Norm-Product Belief Propagation: Primal-Dual Message-Passing for Approximate Inference

Tamir Hazan and Amnon Shashua

***Abstract*—Inference problems in graphical models can be represented as a constrained optimization of a free energy function. In this paper we treat both forms of probabilistic inference, estimating marginal probabilities of the joint distribution and finding the most probable assignment, through a unified message-passing algorithm architecture. In particular we generalize the Belief Propagation (BP) algorithms of sum-product and max-product and tree-rewaighted (TRW) sum and max product algorithms (TRBP) and introduce a new set of convergent algorithms based on "convex-free-energy" and Linear-Programming (LP) relaxation as a zero-temprature of a convex-free-energy. The main idea of this work arises from taking a general perspective on the existing BP and TRBP algorithms while observing that they all are reductions from the basic optimization formula of $f + \sum_i h_i$ where the function f is an extended-valued, strictly convex but non-smooth and the functions h_i are extended-valued functions (not necessarily convex). We use tools from convex duality to present the "primal-dual ascent" algorithm which is an extension of the Bregman successive projection scheme and is designed to handle optimization of the general type $f + \sum_i h_i$. We then map the fractional-free-energy variational principle for approximate inference onto the optimization formula above and introduce the "norm-product" message-passing algorithm. Special cases of the norm-product include sum-product and max-product (BP algorithms), TRBP and NMPLP algorithms. When the fractional-free-energy is set to be convex (convex-free-energy) the norm-product is globally convergent for the estimation of marginal probabilities and for approximating the LP-relaxation. We also introduce another branch of the norm-product which arises as the "zero-temerature" of the convex-free-energy which we refer to as the "convex-max-product". The convex-max-product is convergent (unlike max-product) and aims at solving the LP-relaxation.**

***Index Terms*—Approximate inference, Bethe free energy, Bregman projection, convex free energy, dual block ascent, Fenchel duality, graphical models, linear programming (LP) relaxation, Markov random fields (MRF), maximum a posteriori probability (MAP) estimation, max-product algorithm, sum-product algorithm,**

I. Introduction

PROBABISITIC graphical models present a convenient and popular tool for reasoning about complex distributions. The graphical model reflects the way the complex distribution $p(x_1, ..., x_n)$ factors into a product of potential functions, each defined over a small number of variables, and referred to as *factors*. A graphical model, which defined in terms of *factor graphs*, represents the incidence between factors and the variables by a bipartite graph with one set of nodes corresponding to the variables of the joint distribution and another set of nodes standing for the factors. An edge exists between a variable node and a factor node if the variable is contained in the set of variables represented by the factor. In many applications of interest the factor graph is sparse. In other words, in the modeling of the joint behavior of a set of interacting variables it is often the case that only a small subset of variables interact directly. For example, in the domain of image processing, if we think of each pixel as a variable in a joint distribution over all image pixels then, typically the intensity value of a single pixel will depend most strongly on neighboring pixels in the image, rather than on those at a distant location. Without the local interaction assumption, i.e., if each variable interacts directly with all other variables, then the inference of the joint behavior would be a hopeless task.

Problems involving *inference* using graphical models comes up in a wide range of applications covering a variety of disciplines. Those include digital communications (error correcting codes [13]), computer vision [55], medical diagnosis [25], protein folding [68], computer graphics [14], [9], clustering [49], as well as other broad disciplines which include signal processing, artificial intelligence and statistical physics [15], [27].

Probabilistic inference comes in two distinct forms and typically involve two slightly different algorithmic thrusts. One form of inference task is to obtain one global state of the joint distribution that is most probable, i.e., find the values of $x_1, ..., x_n$ which maximizes $p(x_1, ..., x_n)$. This form of inference is typically referred to as the *maximal a-posteriori* assignment, or in its abbreviated form, the MAP assignment. The second type of inference has the objective of obtaining marginal probabilities for some subset of variables given evidence (value of) about other variables. For example, if $x_i \in \{1, ..., n_i\}$ then $p(x_i)$ comes out of summing exponentially many elements $\sum_{\{x_1,...,x_n\}\setminus x_i} p(x_1, ..., x_n)$ resulting in the likelihood of x_i to obtain each of its possible n_i values. In this paper, we will focus on *both* inference problems with the objective of introducing a unifying algorithmic thrust.

Exact inference is NP-hard [50], thus introducing the need to derive algorithms for *approximate inference*. One of the most popular class of methods for inference over (factor) graphs are message-passing algorithms which pass messages along the edges of the factor graph until convergence is reached. The belief-propagation (BP) algorithms [44] come in two variations: the *sum-product* algorithm for computing marginal probabilities and the *max-product* algorithm for computing the MAP assignment. Citing [69], the centrality of inference using graphical models and the utility of the BP algorithms for solving them is reflected in the fact that

Manuscript resides in arXiv:0903.3127 submitted March 18, 2009 and revised June 16, 2009.

T. Hazan and A. Shashua are with the School of Engineering and Computer Science, Hebrew University of Jerusalem, Jerusalem 91904, Israel (e-mail: tamir@cs.huji.ac.il; shashua@cs.huji.ac.il).

equivalent or very similar message-passing algorithms have been independently derived under different disciplines. Those include the Viterbi algorithm [59], Gallager's sum-product algorithm for decoding low-density parity check codes [16], the turbo-decoding algorithm [3], the Kalman filter for signal processing [28], and the transfer-matrix approach in statistical mechanics [1].

The BP algorithms are exact, i.e., the resulting marginal probabilities and the MAP assignments are the correct ones, when the factor graph is free of cycles — a state of affairs that considerably limits the application of those algorithms to solve real world problems. Nevertheless, an intriguing feature of BP, which most likely is the source for its great popularity, is that it is well-defined and often gives surprisingly good approximate results for graphical models with cycles. However, in this context there are no convergence guarantees (except under some special cases [56], [41]) and the algorithms fail to converge in many cases of interest.

During the past decade there has been much progress in putting forward a framework for approximate inference using variational principles. It has been shown that the fixed-points of the sum-product algorithm (for estimating marginal probabilities) correspond to the fixed-points of a constrained energy function called the Bethe free energy [69]. The free energy arises from the expansion of the KL-divergence between the input distribution and its product form. The Bethe approximation replaces the entropy term in the free energy by the Bethe entropy. The investigation of the stationary points of the Bethe free energy yields conditions for convergence of BP [20], and lower bounds for the free energy in some special cases [54]. These lower bounds are based on the loop calculus framework which considers the Bethe free energy as a first order approximation for the free energy [8]. The Bethe free energy is exact for factor graphs without cycles, as well as convex over the set of constraints (representing validity of marginals). When the factor graph has cycles the Bethe energy is non-convex and the BP algorithms may fail to converge. Although it is possible to derive convergent algorithms to a local minima of the Bethe function [70], [22] the computational cost is large and thus has not gained popularity.

To overcome the difficulty with the non-convexity of the Bethe approximation, several authors have introduced a class of approximations known as *convex free energies* which are convex over the set of constraints for any factor graph. An important member of this class is the tree-reweighted (TRW) free energy which consists of a linear combination of free energies defined on spanning trees of the factor graph [61]. It is notable that for this specific member of convex free energies a convergent message-passing algorithm, applicable to pairwise factors only, has been recently introduced [17]. However, a convergent message passing algorithm for the general class of convex free energies is still lacking. The existing algorithms either employ damping heuristics to ensure convergence in practice [62] or focus on a sub-class of free energies where the entropy term is a positive combination of joint entropies [22].

The MAP assignment problem has been shown to be approximated by a Linear-Programming (LP) relaxation scheme [63] with message-passing algorithmic attempts as a solution [31], [65], [18], [66], [37]. Some of these attempts guarantee convergence only under special cases (such as binary variables), [31], [65]. Others, such as [18], arises as a special case of our algorithm. We refer to [37] for detailed account on the connections between these message-passing algorithms. A double-loop of message passing using a proximal minimization technique proposed recently by [45] is convergent but at a considerable computational expense. Dual decomposition techniques were recently proposed [30], [33], which are related to dual subgradient methods for the LP relaxation.

In this paper, we derive a class of approximate inference message-passing algorithms, which we call *norm-product* algorithms, using the notion of free-energy approximation. The norm-product is an inference engine covering both the estimation of marginal probabilities and the MAP assignment. When the Bethe free energy is used as a substitution for the free-energy, the norm-product reduces to the sum-product and max-product algorithms where the latter emerges as a "zero temperature" version of the former. When a convex-free-energy is used the norm-product becomes a convergent family of algorithms along three strains: (i) a globally convergent algorithm, which we call *convex-sum-product*, for estimating marginal probabilities, (ii) a locally convergent algorithm emerging as a zero-temperature version of the former strain, we call *convex-max-product*, for estimating the MAP assignment, and (iii) a globally convergent algorithm for the LP-relaxation problem.

The convex-sum-product algorithm was published in [19] with only a brief sketch of the detailed derivation. In this paper we have chosen to put a large amount of material in appendices. Due to the complexity of the presented material and the extensive use of modern optimization infrastructure, the body of the paper contains the main "storyline", statements and algorithms whereas the detailed proofs and the required mathematical infrastructure are contained in appendices.

II. Notations, Problem Setup and Background

Let $x_1, ..., x_n$ be the realizations of n discrete random variables where the range of the $i'th$ random variable is $\{1, ..., n_i\}$, i.e., $x_i \in \{1, ..., n_i\}$. We consider a joint distribution $p(x_1, ..., x_n)$ and assume that it factors into a product of non-negative functions (potentials):

$$p(x_1, ..., x_n) = \frac{1}{Z} \prod_{i=1}^{n} \phi_i(x_i) \prod_{\alpha=1}^{m} \psi_\alpha(\mathbf{x}_\alpha), \qquad (1)$$

where the functions $\phi_i(x_i)$ represent "local evidence" or prior data on the states of x_i, and the functions $\psi_\alpha(\mathbf{x}_\alpha)$ have arguments $\mathbf{x}_\alpha$ that are some subset of $\{x_1, ..., x_n\}$ and Z is a normalization constant, typically referred as the *partition function*. For example, $p(x_1, x_2, x_3) = (1/Z)\psi_{23}(x_2, x_3)\psi_{13}(x_1, x_3)$ has two factors with indices $\alpha_1 = \{2, 3\}, \alpha_2 = \{1, 3\}$ and $\mathbf{x}_{\alpha_1} = \{x_2, x_3\}, \mathbf{x}_{\alpha_2} = \{x_1, x_3\}$, and uniform local evidence $\phi_i(x_i) = 1$ for every $i = 1, 2, 3$ and every x_i.

The factorization structure above defines a *hypergraph* whose nodes represent the n random variables and the subsets of variables $\mathbf{x}_\alpha$ correspond to its hyperedges. For example, if all factor functions are defined on pairs of random variables then the factorization is represented by a graph. A convenient way to represent hypergraphs is by a bipartite graph with one set of nodes corresponding to the original nodes of the hypergraph and the other set corresponds to its hyperedges. In the context of graphical models such a bipartite graph representation is referred to as a *factor graph* [35] with *variable nodes* representing $\phi_i(x_i)$ and a *factor node* for each function $\psi_\alpha(\mathbf{x}_\alpha)$. An edge connects a variable node i with factor node α if and only if $x_i \in \mathbf{x}_\alpha$, i.e., x_i is an argument of ψ_α. We adopt the terminology where $N(i)$ stands for all factor nodes that are neighbors of variable node i, i.e., all the nodes α for which $x_i \in \mathbf{x}_\alpha$, and $N(\alpha)$ stands for all variable nodes that are neighbors of factor node α.

We shall focus on the two inference tasks of computing marginal probabilities and maximum a-priori (MAP) assignment. The computation of the marginal probabilities $p(x_i) = \sum_{\mathbf{x}\setminus x_i} p(\mathbf{x})$ and $p(\mathbf{x}_\alpha) = \sum_{\mathbf{x}\setminus \mathbf{x}_\alpha} p(\mathbf{x})$, requires the summation over the states of all the variable nodes not in x_i or $\mathbf{x}_\alpha$ respectively. This computation is generally hard because it may require summing up exponentially large number of terms — thus one seeks efficient ways or approximate solutions for the marginals. The MAP assignment is the task of finding a state for each x_i that brings the maximal value to the joint probability $p(x_1, ..., x_n)$.

The belief-propagation (BP) algorithms, known as sum-product and max-product, are two algorithms for computing marginal probability and MAP assignment, respectively, that can be described in terms of operations on a factor graph. As already mentioned in the introduction, the BP algorithms will deliver the correct inference, i.e. are exact, if the factor graph has no cycles, but are still well defined and often provide good approximate results when the factor graph has cycles.

The BP algorithms are defined in terms of *messages* between variable and factor nodes. The message $m_{\alpha\to i}(x_i)$ from factor node α to variable node i, and the opposite direction message $n_{i\to\alpha}(x_i)$, is a vector over the states of x_i. In the sum-product algorithm those have the following form:

$$\begin{aligned} m_{\alpha\to i}(x_i) &= \sum_{\mathbf{x}_\alpha\setminus x_i} \psi_\alpha(\mathbf{x}_\alpha) \prod_{j\in N(\alpha)\setminus i} n_{j\to\alpha}(x_j) \\ n_{i\to\alpha}(x_i) &\propto \phi_i(x_i) \prod_{\beta\in N(i)\setminus\alpha} m_{\beta\to i}(x_i) \end{aligned}$$

The $\propto$ indicates that one can normalize the vector. The messages $n_{i\to\alpha}(x_i)$ are usually initialized to the uniform vector. Upon convergence of the message-passing scheme the marginal probabilities $p(x_i)$ and $p(\mathbf{x}_\alpha)$ can be expressed in terms of a "pseudo-distribution", also known as *beliefs*, $b_i(x_i)$ and $b_\alpha(\mathbf{x}_\alpha)$ defined below:

$$\begin{aligned} b_i(x_i) &\propto \phi_i(x_i) \prod_{\alpha\in N(i)} m_{\alpha\to i}(x_i) \\ b_\alpha(\mathbf{x}_\alpha) &\propto \psi_\alpha(\mathbf{x}_\alpha) \prod_{j\in N(\alpha)} n_{j\to\alpha}(x_j) \end{aligned}$$

When the factor graph has no cycles the messages converge and the beliefs correspond to the marginal probabilities. When the factor graph has cycles there is no convergence guarantee and, regardless of convergence, the recovered beliefs provide only an approximation to the marginal probabilities.

In the max-product algorithm the messages $m_{\alpha\to i}(x_i)$ are slightly altered:

$$m_{\alpha\to i}(x_i) = \max_{\mathbf{x}_\alpha\setminus x_i} \left\{ \psi_\alpha(\mathbf{x}_\alpha) \prod_{j\in N(\alpha)\setminus i} n_{j\to\alpha}(x_j) \right\},$$

while $n_{i\to\alpha}(x_i)$ remain as in the sum-product algorithm. The MAP assignment can be recovered from the beliefs $b_i(x_i)$ when the factor graph is a tree. In such a case, the MAP assignment of x_i corresponds to the index of highest entry of $b_i(x_i)$. In general convergence is not guaranteed, and the MAP assignment can be recovered only for specific problems, [64], [2], [24], [47].

A. Inference using a Variational Principle

The BP algorithms apply to tree-structured factor graphs yet are well defined for general factor graphs but without convergence or accuracy guarantees. The variational principle approach, described below, is a decade long effort at providing an extended platform from which old, i.e., BP algorithms, and new (preferably convergent) algorithms can emerge.

The variational approach seeks a distribution $p(x_1, ..., x_n)$ that is as close as possible, in relative entropy terms, to the product potentials $\phi_i(x_i)$ and $\psi_\alpha(\mathbf{x}_\alpha)$. Expanding the KL-divergence $D(\mathbf{p} \,||\, \mathbf{q}) = \sum_x p(x)\ln(p(x)/q(x))$ between two nonnegative vectors results in:

$$D(\mathbf{p} \,||\, \prod_i \phi_i \prod_\alpha \psi_\alpha) = F(\mathbf{p}),$$

where $F(\mathbf{p})$ is the so called Gibbs-Helmholtz free-energy:

$$\begin{aligned} F(\mathbf{p}) &= \sum_{i,x_i} \theta_i(x_i)p(x_i) + \sum_{\alpha,\mathbf{x}_\alpha} \theta_\alpha(\mathbf{x}_\alpha)p(\mathbf{x}_\alpha) - H(\mathbf{p}) \\ &= E(\mathbf{p}) - H(\mathbf{p}) \qquad (2) \end{aligned}$$

The term $H(\mathbf{p}) = -\sum_{\mathbf{x}} p(\mathbf{x})\ln p(\mathbf{x})$ is the entropy and $\theta_i = -\ln\phi_i$ and $\theta_\alpha = -\ln\psi_\alpha$. The linear term $E(\mathbf{p})$ is often referred to as the *energy* term. By minimizing $F(\mathbf{p})$ over the probability simplex $\mathcal{P} = \{\mathbf{p} : \mathbf{p} \geq 0, \sum_{\mathbf{x}} p(\mathbf{x}) = 1\}$ we get back the probability distribution defined in eqn. 1, as the optimal argument $\mathbf{p}^* = \text{argmin}_{\mathbf{p}\in\mathcal{P}} F(\mathbf{p})$, and minus the log of the normalization, or equivalently the partition function, as the value at the minimum:

$$\mathbf{p}^* = \frac{1}{Z}\prod_{i=1}^n \phi_i(x_i) \prod_{\alpha=1}^m \psi_\alpha(\mathbf{x}_\alpha), \qquad -\ln Z = F(\mathbf{p}^*).$$

Since $F(\mathbf{p})$ is strictly convex and the simplex constraints are convex, the minimum is unique. So far we have not gained anything because the entropy $H(\mathbf{p})$ is computationally intractable since its evaluation is exponential in n, and satisfying the probability simplex constraints is intractable as well. The variational methods are based on a tractable approximation to the free-energy $F(\mathbf{p})$ by (i) approximating the entropy term $H(\mathbf{p})$ by a combination of local entropies over marginal probabilities $p(x_i), p(\mathbf{x}_\alpha)$, and (ii) by approximating the probability simplex constraints by the so called "marginal consistency" constraints.

In approximate inference, the true marginal distributions $p(x_i)$ and $p(\mathbf{x}_\alpha)$ are replaced by "beliefs" $b_i(x_i)$ and $b_\alpha(\mathbf{x}_\alpha)$ which form a "pseudo distribution" in the sense that the beliefs might not necessarily arise as marginals of some distribution $p(x_1, ..., x_n)$. The probability simplex constraints are replaced by *marginal consistency* constraints $\mathbb{L}(G)$ defined below:

$$\mathbb{L}(G) = \left\{ \mathbf{b} = \{\mathbf{b}_i, \mathbf{b}_\alpha\} : \begin{array}{l} \sum_{\mathbf{x}_\alpha \setminus x_i} b_\alpha(\mathbf{x}_\alpha) = b_i(x_i) \quad \forall i, \alpha \in N(i) \\ b_\alpha(\mathbf{x}_\alpha) \geq 0, \ \sum_{\mathbf{x}_\alpha} b_\alpha(\mathbf{x}_\alpha) = 1 \quad \forall \alpha \end{array} \right.$$

The entropy approximation $\tilde{H}(b)$ as a function of the beliefs is known as *fractional entropy* and has the form:

$$\sum_\alpha \bar{c}_\alpha H(\mathbf{b}_\alpha) + \sum_i \bar{c}_i H(\mathbf{b}_i), \tag{3}$$

where the joint entropy $H(\mathbf{b}_\alpha) = -\sum_{\mathbf{x}_\alpha} b_\alpha(\mathbf{x}_\alpha) \ln b_\alpha(\mathbf{x}_\alpha)$ and the local entropy $H(\mathbf{b}_i) = -\sum_{x_i} b(x_i) \ln b(x_i)$.

For factor-graphs without cycles, the setting of $\bar{c}_\alpha = 1$ and $\bar{c}_i = 1 - d_i$ where d_i is the degree of the variable node associated with x_i in the factor graph, renders the approximation $\tilde{H}$ to be *exact* and equal[1] to the entropy H. Such an approximation is known as the *Bethe* entropy:

$$H_{bethe}(\mathbf{b}) \overset{def}{=} \sum_\alpha H(\mathbf{b}_\alpha) + \sum_i (1 - d_i) H(\mathbf{b}_i).$$

Moreover, in the case of a tree, the marginal consistency constraints $\mathbb{L}(G)$ are equal to the probability simplex constraints, thus making the constrained *Bethe free energy* problem

$$\min_{\mathbf{b} \in \mathbb{L}(G)} \sum_{i, x_i} \theta_i(x_i) b_i(x_\alpha) + \sum_{\alpha, \mathbf{x}_\alpha} \theta_\alpha(\mathbf{x}_\alpha) b_\alpha(\mathbf{x}_\alpha) - H_{bethe}(\mathbf{b}),$$

a convex optimization producing the true marginals $b_i(x_i) = p(x_i)$ and $b_\alpha(\mathbf{x}_\alpha) = p(\mathbf{x}_\alpha)$. The constrained optimization is defined in terms of beliefs only and is therefore computationally tractable. However, if the factor graph has cycles, the minimizer of the constrained Bethe free energy is not guaranteed to correspond to the true marginals $p(x_i)$, $p(\mathbf{x}_\alpha)$, and not even realizable as a true distribution. Therefore, for general factor graphs, the Bethe free energy optimization approach finds an approximation to the true marginal probabilities. From the optimization point of view, the Bethe free energy is strictly convex in the intersection of constraints when the factor graph is a tree. When the factor graph has cycles the Bethe energy is non-convex and although it is possible to derive convergent algorithms to local minima of the Bethe function [70], [22] the computational cost is large and thus has not gained popularity.

What makes the Bethe free energy optimization interesting is the observation, first elucidated by [69], that when the sum-product algorithm converges then it does so to a stationary point of the constrained Bethe free energy, i.e., fixed-points of the algorithm correspond to stationary points of the variational problem. This does not mean that the sum-product algorithm descends on the Bethe free energy (in fact it does not), but that near a fixed point things start to behave well. The significance of the observation is that it ties the popular sum-product algorithm with a specific variational principle and moreover it suggests a framework for seeking natural generalizations of the Bethe approximation with their associated message-passing algorithms.

[1]in this case the joint probability can be expressed solely in terms of the marginals: $p(x_1, ..., x_n) = \prod_\alpha p(\mathbf{x}_\alpha) / \prod_i p(x_i)^{d_i - 1}$. Expanding $H(p)$ produces the Bethe entropy approximation.

Generalizations of the Bethe free energy move along two thrusts. The first employs better (higher-order) approximations to the entropy and higher-order constraints beyond the marginal consistency constraints to better approximate the full probability simplex constraints. This effort includes Kikuchi free energy, region graphs and other hyper-graph based methods [69], [29]. The second thrust looks for convergence guaranteed message-passing algorithms by extending the Bethe free energy to form a wider class of functions, known as *convex free energies*, which are convex in the intersection of marginal consistency constraints. In this paper we focus on the second thrust. The inclusion of Kikuchi approximations and region graphs is a natural extension to the results we introduce in this paper but for the sake of clarity we leave it outside the current scope.

The fractional entropy eqn. 3 can be set to form a family of concave approximations. The set of sufficient conditions for an entropy approximation of the type of eqn. 3 to be concave over the set of constraints was introduced in [21], [65] and take the following form:

Definition 1 (Concave Entropy Approximation): An approximate entropy term of the form eqn. 3 is strictly concave over the set of marginal consistency constraints if there exists $c_i, c_{i\alpha} \geq 0$ and $c_\alpha > 0$ such that $\bar{c}_\alpha = c_\alpha + \sum_{i \in N(\alpha)} c_{i\alpha}$ and $\bar{c}_i = c_i - \sum_{\alpha \in N(i)} c_{i\alpha}$. The approximate entropy $\tilde{H}(\mathbf{b})$ becomes:

$$\sum_\alpha c_\alpha H(\mathbf{b}_\alpha) + \sum_i c_i H(\mathbf{b}_i) + \sum_{i, \alpha \in N(i)} c_{i\alpha} (H(\mathbf{b}_\alpha) - H(\mathbf{b}_i)). \tag{4}$$

The entropy approximation $\tilde{H}(\mathbf{b})$ includes the Bethe approximation when $c_\alpha = 1, c_i = 1 - d_i$ and $c_{i\alpha} = 0$ but it is guaranteed to be strictly concave for any setting of the parameters where $c_i, c_{i\alpha} \geq 0$ and $c_\alpha > 0$. An important member of this class is the "tree-reweighted" (TRW) approximation [62] where $\bar{c}_\alpha$ is equal to a weighted combination of spanning trees of the original graph (all factors are pairwise and thus α represents an edge) which pass through α. In Appendix D we describe a number of concave settings of $\tilde{H}$ including TRW and other heuristic settings. The convex-free-energy variational program becomes:

$$\min_{\mathbf{b} \in \mathbb{L}(G)} \sum_{i, x_i} b_i(x_i) \theta_i(x_i) + \sum_{\alpha, \mathbf{x}_\alpha} \theta_\alpha(\mathbf{x}_\alpha) b_\alpha(\mathbf{x}_\alpha) - \tilde{H}(\mathbf{b}). \tag{5}$$

The global minimizer $\mathbf{b}^*$ of the convex-free-energy program above is an approximation to the marginal probabilities due to (i) the term $\tilde{H}$ is an approximation to the entropy of the distribution and its quality depends on how the parameters $c_\alpha, c_i, c_{i\alpha}$ are set and on the structure of the factor graph, and (ii) due to the fact that the marginal consistency constraints

$\mathbb{L}(G)$ approximate the probability simplex constraints, there is no guarantee that in general $\mathbf{b}^*$ form a distribution, i.e., the marginal estimations $b_\alpha^*(\mathbf{x}_\alpha)$ and $b_i^*(x_i)$ might not arise from any probability distribution over $x_1, ..., x_n$.

The only guarantees we have is that if the factor graph has no cycles then the marginal probabilities are exact and if $\tilde{H}$ is strictly concave then it should be possible to generate a convergent message-passing algorithm (unlike BP algorithms which are not generally convergent).

We move next to the MAP assignment task where one seeks a vector $\mathbf{x}^*$ which maximizes the product of potentials, or equivalently minimizes the energy

$$\operatorname*{argmax}_{\mathbf{x}} \prod_i \phi_i(x_i) \prod_\alpha \psi_\alpha(\mathbf{x}_\alpha) = \operatorname*{argmin}_{\mathbf{x}} \sum_i \theta_i(x_i) + \sum_\alpha \theta_\alpha(x_\alpha).$$

Described as a variational principle program, the MAP assignment problem is equivalent to the linear program whose variables corresponds to distribution $p(x_1, ..., x_n)$ with exponential many elements:

$$\min_{p(\mathbf{x}) \geq 0, \sum_{\mathbf{x}} p(\mathbf{x}) = 1} \sum_{i, x_i} \theta_i(x_i) p(x_i) + \sum_{\alpha, \mathbf{x}_\alpha} \theta_\alpha(\mathbf{x}_\alpha) p(\mathbf{x}_\alpha).$$

The optimization of a linear function over the probability simplex yields an optimal solution $\mathbf{p}^*$ in an extreme point of the probability simplex, namely $\mathbf{p}^*$ is a zero-one distribution. In particular $p^*(\mathbf{x}^*) = 1$ and for every $\mathbf{x} \neq \mathbf{x}^*$ holds $p^*(\mathbf{x}) = 0$.

An approximation can be obtained by approximating the marginal probabilities $p(x_i)$ and $p(\mathbf{x}_\alpha)$ with beliefs $b_i(x_i)$ and $b_\alpha(\mathbf{x}_\alpha)$ which are not guaranteed to correspond to a true distribution over $x_1, ..., x_n$.

$$\min_{\mathbf{b}_i, \mathbf{b}_\alpha \in \mathbb{L}(G)} \sum_{i, x_i} \theta_i(x_i) b_i(x_i) + \sum_{\alpha, \mathbf{x}_\alpha} \theta_\alpha(\mathbf{x}_\alpha) b_\alpha(\mathbf{x}_\alpha) \quad (6)$$

If the minimizer of the LP-relaxation problem comes out without ties, i.e., the marginal vectors $b_i(x_i)$ have a single maximal entry, then the MAP assignment readily emerges from the LP-relaxed solution. This LP-relaxed problem can be solved using off-the-shelf LP solvers but the key problem with standard LP solvers, however, is that they do not use the graph structure explicitly and thus are sub-optimal in terms of computational efficiency. An empirical study found the message-passing LP-solvers, e.g. max-TRBP, to be superior to the CPLEX solver, a commercial LP solver that implements different techniques for solving LP, such as primal and dual simplex solvers, network solvers, primal-dual barrier solver for sparse problem, and sifting techniques executing sequences of LP subproblems [67].

The relaxed LP problem of eqn. 6 has been widely studied in the literature in the context of message-passing algorithms. Special cases of these LP-relaxations were used for constraints satisfaction [48], [34]. The general form in eqn. 6 was studied using tree decompositions in [63], [30], as well as dual decomposition [32], [33], and dual block coordinate ascent [66], [18], [52]. A general framework for these recent developments is described in [37]. Since the LP energy is not strictly convex, convergence to the global minimum is a challenge, since eqn. 6 usually corresponds to a non-smooth dual. In this case a dual block coordinate ascent can lead to a corner in the dual objective, which is a non-optimal stationary point.

An alternative class of methods are based on a (strictly) convex relaxation approach. There are two notable recent examples in this class: one using a proximal minimization technique where the convex term is a weighted KL-divergence measure between the sought-after belief vector and the one from the previous iteration [45]. The proximal minimization approach involves a double-loop of message passing iterations and is guaranteed to converge to the global optimum of eqn. 6. The second approach, the one we follow in this paper, is to make eqn. 6 the "zero temperature" of the perturbed problem:

$$\min_{\mathbf{b} \in \mathbb{L}(G)} \sum_{i, x_i} \theta_i(x_i) b_i(x_i) + \sum_{\alpha, \mathbf{x}_\alpha} \theta_\alpha(\mathbf{x}_\alpha) b_\alpha(\mathbf{x}_\alpha) - \epsilon \tilde{H}(\mathbf{b}), \quad (7)$$

by taking $\epsilon \to 0$. This approach was used in decoding low-density parity-check codes [60]. It was also used for LP-relaxations, to derive a non-convergent max-product like algorithm [65], and for applying an iterative proportional fitting type algorithm [26].

This concludes the necessary background to inference within the framework of variational principle. The variational problem we will work on next is eqn. 7. We will derive a convergent message-passing algorithm called the norm-product. When the parameters of $\tilde{H}$ are set to the Bethe approximation the algorithm reduces to the sum-product (when $\epsilon = 1$) or the max-product (when $\epsilon = 0$). When $\tilde{H}$ is concave and $\epsilon = 1$ the norm-product becomes a globally convergent message-passing algorithm, referred to as convex-sum-product, for approximating marginal probabilities. When $\epsilon = 0$ we obtain a convergent form of max-product we call convex-max-product and when $\epsilon \to 0$ we obtain an approximation (with proven bounds) to the LP-relaxation solution.

III. The Norm-Product Belief Propagation Algorithm

We seek an algorithm for minimizing the inference variational eqn. 7 with the following properties: (i) if the entropy approximation term $\tilde{H}$ is strictly concave, i.e., eqn. 7 is a convex-free-energy, the algorithm will be convergent for all $\epsilon \geq 0$ and will converge to the global optimum when $\epsilon > 0$, (ii) the algorithm will remain well defined when $\tilde{H}$ is non-convex (such as Bethe-free-energy and other fractional entropy approximations) and exhibit the property that fixed points of the algorithm coincide with stationary points of eqn. 7, and (iii) the algorithm uses the graph structure inherent sparseness, i.e., is defined by a message-passing architecture on the underlying factor-graph. In other words, like the BP-algorithms, our scheme should be sending messages between variable and factor nodes of the factor graph.

We will first take a detour and derive a general framework for minimizing problems of the type

$$\min_{\mathbf{b}} f(\mathbf{b}) + \sum_{i=1}^n h_i(\mathbf{b}) \quad (8)$$

$f(\mathbf{b})$ is a strictly convex, non-smooth, extended-valued function of the type $f(\mathbf{b}) = \hat{f}(\mathbf{b}) + \delta_B(\mathbf{b})$ where $\hat{f}$ is essentially smooth and $\delta_{\mathcal{B}}$ is the indicator function of the affine set $\mathcal{B} = \{\mathbf{b} \ : \ A\mathbf{b} = \mathbf{c}\}$, namely, $\delta_{\mathcal{B}}(\mathbf{b}) = 0$ if $\mathbf{b} \in \mathcal{B}$ and ∞ otherwise. The functions $h_i(\mathbf{b})$ are convex extended-valued functions (see Appendix A on mathematical background). In Appendix B we derive the following "primal-dual" block ascent algorithm which is guaranteed to converge to the global minimizer of eqn. 8:

Algorithm 1 (Primal-Dual Ascent): Let $f(\mathbf{b}) = \hat{f}(\mathbf{b}) + \delta_{\mathcal{B}}(\mathbf{b})$ where $\hat{f}(\mathbf{b})$ is strictly convex, essentially smooth extended-valued function, and let $h_i(\mathbf{b})$ be convex extended-valued functions. Initialize $\boldsymbol{\lambda}_1 = \mathbf{0}, ..., \boldsymbol{\lambda}_n = \mathbf{0}$.

```
1) Repeat until convergence:
2) For $i = 1, ...n$:
    a) $\boldsymbol{\mu}_i = \sum_{j \neq i} \boldsymbol{\lambda}_j$
    b) $\mathbf{b}^* = \operatorname{argmin}_{\mathbf{b} \in dom(h_i) \cap dom(f)} \{f(\mathbf{b}) + h_i(\mathbf{b}) + \mathbf{b}^\top \boldsymbol{\mu}_i\}$
    c) $\boldsymbol{\lambda}_i = -\boldsymbol{\mu}_i - \nabla \hat{f}(\mathbf{b}^*) + A^\top \boldsymbol{\sigma}$ where $\boldsymbol{\sigma}$ is arbitrary.
Output $\mathbf{b}^*$.
```

The vectors $\boldsymbol{\lambda}_i$ and $\boldsymbol{\mu}_i$ are messages passed along edges of a bipartite graph with n (function) nodes corresponding to the n functions $h_i(\mathbf{b})$ and m (variable) nodes corresponding to the dimension of $\mathbf{b}$. Function node i sends the m coordinates of vector $\boldsymbol{\lambda}_i$ to the m variable nodes. Variable node j sends the j'th coordinate of vectors $\boldsymbol{\mu}_1, ..., \boldsymbol{\mu}_n$ to the n functions nodes. The algorithm iteratively optimizes with respect to the indexes $i \in \{1, ..., n\}$, stopping when it does not change the beliefs $\mathbf{b}^*$, thus the network proceeds in an almost cyclic update policy. The algorithm fits well with a graphical model architecture in the sense that if $h_i(\mathbf{b})$ depends only on a small subset $N(i)$ of coordinates from $\mathbf{b}$, then $\boldsymbol{\lambda}_{i\beta} = 0$ for every $\beta \notin N(i)$ (and therefore need not be updated):

Claim 1: Assume variables $\mathbf{b}$ are indexed by $\{1, ..., m\}$ and $h_i(\mathbf{b})$ depends only on small subset of variables indexed by $N(i) \subset \{1, ..., m\}$ and let $\boldsymbol{\lambda}_i = \{\boldsymbol{\lambda}_{i,\alpha}\}$. Then, $\boldsymbol{\lambda}_{i,\beta} = \mathbf{0}$ for all $\beta \notin N(i)$.

The claim and its proof can be found in Appendix B. For those familiar with successive projection schemes, in the particular case when $f(\mathbf{b}) = \hat{f}(\mathbf{b})$, i.e., is strictly convex and essentially smooth, and $h_i(\mathbf{b}) = \delta_{C_i}(\mathbf{b})$ (the indicator function of convex set C_i), the update step (b) for Algorithm 1 is a "Bregman" projection [6] of the vector $\boldsymbol{\mu}_i$ onto the convex set C_i. In that case, following some algebraic manipulations (such as eliminating $\boldsymbol{\mu}_i$ among other manipulations) the scheme (with $A = 0$) reduces to the well known Dykstra [12] (also goes under different names such as Hildreth, Bregman, Csiszar, Han) successive projection algorithm which has its origins in the work of Von-Neumann [43]. Further historical details can be found in Appendix B.

Another useful property of the algorithm that it is well defined for non-convex primal energies. Specifically, we can establish the following result:

Claim 2: Consider Algorithm 1 for Legendre-type function $f(\mathbf{b})$ and non-convex continuously differentiable functions $h_i(\mathbf{b})$ restricted to the affine domain $\{\mathbf{b} \ : \ A_i\mathbf{b} = \mathbf{c}_i\}$, and assume $\mathbf{b}^*$ in step (c) is in the interior of $dom(f)$. Then, fixed-points of the algorithm coincide with stationary points of the non-convex program $f(\mathbf{b}) + \sum_i h_i(\mathbf{b})$.

The proof can be found in Appendix B-A. The result states that when h_i are non-convex but defined over an affine domain the algorithm is no longer convergent, but if it does converge it will do so to a stationary point of the optimization problem. This property of the algorithm extends the result of [69] about the behavior of the sum-product algorithm: if it converges, then it converges to a stationary point of the Bethe free-energy.

The inference variational problem presented in eqn. 7 is embedded into the general template of eqn. 8 as follows:

$$\min_{\mathbf{b}} f_\epsilon(\mathbf{b}) + \sum_{i=1}^n h_{\epsilon,i}(\mathbf{b}) \tag{9}$$

where $f_\epsilon(\mathbf{b}) = \hat{f}_\epsilon(\mathbf{b}) + \delta_S(\mathbf{b})$ with S being the set of $\{\mathbf{b} = \{\mathbf{b}_\alpha\} \ : \ b_\alpha(\mathbf{x}_\alpha) \in \mathcal{P}\}$ where $\mathcal{P}$ is the probability simplex (arrays that are non-negative and sum to one), and $\hat{f}_\epsilon$ is defined below:

$$\hat{f}_\epsilon(\mathbf{b}_\alpha) = \sum_{\alpha,\mathbf{x}_\alpha} \theta_\alpha(\mathbf{x}_\alpha) b_\alpha(\mathbf{x}_\alpha) - \sum_\alpha \epsilon c_\alpha H(\mathbf{b}_\alpha) \tag{10}$$

Note that $dom(f_\epsilon)$ include all $\mathbf{b} \in S$, i.e., $f_\epsilon(\mathbf{b}) = \infty$ for $\mathbf{b} \notin S$. The functions $h_{\epsilon,i}$ are defined below:

$$h_{\epsilon,i}(\mathbf{b}) = \sum_{x_i} \theta_i(x_i) b_i(x_i) - \epsilon c_i H(\mathbf{b}_i) - \sum_{\alpha \in N(i)} \epsilon c_{i\alpha}(H(\mathbf{b}_\alpha) - H(\mathbf{b}_i)), \tag{11}$$

where $dom(h_{\epsilon,i})$ is the affine set consisting of $\mathbf{b}_\alpha$ for every $\alpha \in N(i)$, which live in the probability simplex, i.e. $dom(h_{\epsilon,i}) \subset dom(f_\epsilon)$, and satisfy the marginal consistency constraints $\sum_{\mathbf{x}_\alpha \setminus x_i} b_\alpha(\mathbf{x}_\alpha) = b_i(x_i)$. Note that $\mathbf{b}_i$ are not explicitly included in $\mathbf{b} = \{\mathbf{b}_\alpha\}$, but they are described by the values of which all $\mathbf{b}_\alpha$ in the domain of $h_{\epsilon,i}$ agree upon.

Given the sparse structure of $h_{\epsilon,i}$ then, following Claim 1, we present the entries of $\boldsymbol{\lambda}_i$ according to the factor-graph structure by setting $\boldsymbol{\lambda}_i = \{\lambda_{i,\alpha}(\mathbf{x}_\alpha)\}$ (and likewise $\boldsymbol{\mu}_i$). Step (b) of Algorithm 1 is reduced to finding $\mathbf{b}_\alpha^*$ for all $\alpha \in N(i)$ and step (c) updates $\boldsymbol{\lambda}_{i,\alpha}$ by the rule:

$$\lambda_{i,\alpha}(\mathbf{x}_\alpha) = -\mu_{i,\alpha}(\mathbf{x}_\alpha) - \nabla \hat{f}_\epsilon(b_\alpha^*(\mathbf{x}_\alpha)) + \sigma_\alpha \mathbf{1},$$

for an arbitrary σ_α. If instead of updating $\boldsymbol{\lambda}_{i,\alpha}$ we would update $n_{i\to\alpha}(\mathbf{x}_\alpha) = \exp(-\lambda_{i,\alpha}(\mathbf{x}_\alpha))$ the additive degree of freedom inherent in the choice of $\boldsymbol{\sigma}$ turns into a scaling choice of $n_{i\to\alpha}$.

The derivation process required for embedding the definitions above into the primal dual Algorithm 1 is described in detail in Appendix C. The resulting algorithm, we call *norm-product*, is presented in Fig. 1.

Just as in the BP algorithms, the *message* $m_{\alpha\to i}(x_i)$ from the factor node α to the variable node i is a vector over all possible states of x_i. The message $n_{i\to\alpha}(\mathbf{x}_\alpha)$ from the variable node i to the factor node α is an *array* over all possible states of $\mathbf{x}_\alpha$. The beliefs $b_i(x_i)$, which are the approximations to the marginal probability $p(x_i)$ when $\epsilon = 1$, can be computed from the messages $m_{\alpha\to i}$:

$$b_i(x_i) \propto \left(\phi_i(x_i) \prod_{\alpha \in N(i)} m_{\alpha\to i}(x_i) \right)^{1/\epsilon \hat{c}_i}, \tag{12}$$

Algorithm 2 (Norm-Product Belief Propagation): We are given nonnegative local evidence $\phi_i(x_i)$, and nonnegative arrays $\psi_\alpha(\mathbf{x}_\alpha)$, where $\alpha \subset \{1,...,n\}$. Let $\hat{c}_{i\alpha} = c_\alpha + c_{i\alpha}$ and $\hat{c}_i = c_i + \sum_{\alpha \in N(i)} c_\alpha$.

```
1) Set $n_{i\to\alpha}(\mathbf{x}_\alpha) = 1$ for all $i = 1,...,n$, $\alpha \in N(i)$ and $\mathbf{x}_\alpha$.
2) For $t = 1,2,...$
    a) For $i = 1,...n$ do:
```

$$\forall x_i\ \forall \alpha \in N(i) \quad m_{\alpha\to i}(x_i) = \left(\sum_{\mathbf{x}_\alpha \setminus x_i} \left(\psi_\alpha(\mathbf{x}_\alpha) \prod_{j \in N(\alpha)\setminus i} n_{j\to\alpha}(\mathbf{x}_\alpha)\right)^{1/(\epsilon \hat{c}_{i\alpha})}\right)^{\epsilon \hat{c}_{i\alpha}}$$

$$\forall \alpha \in N(i)\ \forall \mathbf{x}_\alpha \quad n_{i\to\alpha}(\mathbf{x}_\alpha) \propto \left(\frac{\phi_i^{1/\hat{c}_i}(x_i) \prod_{\beta \in N(i)} m_{\beta\to i}^{1/\hat{c}_i}(x_i)}{m_{\alpha\to i}^{1/\hat{c}_{i\alpha}}(x_i)}\right)^{c_\alpha} \left(\psi_\alpha(\mathbf{x}_\alpha) \prod_{j \in N(\alpha)\setminus i} n_{j\to\alpha}(\mathbf{x}_\alpha)\right)^{-c_{i\alpha}/\hat{c}_{i\alpha}}$$

Fig. 1. The norm-product belief propagation algorithm, where the messages $m_{\alpha\to i}(x_i)$ are computed with respect to the $L_{1/\epsilon \hat{c}_{i\alpha}}$ norm. For $c_\alpha = 1, c_i = 1 - d_i, c_{i\alpha} = 0$ it reduces to the belief propagation algorithms, sum-product when $\epsilon = 1$ and max-product when $\epsilon = 0$. Whenever c_α is the weighted number of spanning trees through edge α, and $c_i = 1 - \sum_{\alpha \in N(i)} c_\alpha$ and $c_{i\alpha} = 0$ it reduces to the tree-reweighted belief propagation algorithms (sum-TRBP and max-TRBP). Whenever $c_\alpha > 0, c_i, c_{i\alpha} \geq 0$ the norm-product is guaranteed to converge, and if also $\epsilon > 0$ it converges to the global optimum of the program in eqn. 7.

where $\hat{c}_i$ is defined in Fig. 1. The joint beliefs $b_\alpha(\mathbf{x}_\alpha)$ can be computed from the messages $n_{i\to\alpha}$:

$$b_\alpha(\mathbf{x}_\alpha) \propto \left(\psi_\alpha(\mathbf{x}_\alpha) \prod_{i \in N(\alpha)} n_{i\to\alpha}(\mathbf{x}_\alpha)\right)^{1/\epsilon c_\alpha}. \quad (13)$$

The norm-product algorithm includes the BP algorithms (sum-product and max-product), as well as sum-TRBP [62], max-TRBP [63], and NMPLP [18] as particular cases. These algorithms relate to the simpler form of the norm-product algorithm, when $c_{i\alpha} = 0$. In this setting the messages $n_{i\to\alpha}(\mathbf{x}_\alpha)$ depend solely on the local potentials $\phi_i(x_i)$ and the messages $m_{\beta\to i}(x_i)$. Therefore the messages $n_{i\to\alpha}(\mathbf{x}_\alpha)$ can be written in the compact form $n_{i\to\alpha}(x_i)$, replacing $\mathbf{x}_\alpha$ with x_i. In this case the norm-product algorithm in Fig. 1 takes the form:

$$m_{\alpha\to i}(x_i) = \left(\sum_{\mathbf{x}_\alpha \setminus x_i} \left(\psi_\alpha(\mathbf{x}_\alpha) \prod_{j \in N(\alpha)\setminus i} n_{j\to\alpha}(\mathbf{x}_\alpha)\right)^{1/\epsilon c_\alpha}\right)^{\epsilon c_\alpha}$$

$$n_{i\to\alpha}(x_i) \propto \frac{\left(\phi_i(x_i) \prod_{\beta \in N(i)\setminus\alpha} m_{\beta\to i}(x_i)\right)^{c_\alpha/\hat{c}_i}}{m_{\alpha\to i}(x_i)}$$

When using the norm-product with the Bethe entropy approximation $c_{i\alpha} = 0, c_\alpha = 1, c_i = 1 - d_i$ there holds $\hat{c}_i = 1$ and the algorithm reduces to

$$m_{\alpha\to i}(x_i) = \left(\sum_{\mathbf{x}_\alpha \setminus x_i} \left(\psi_\alpha(\mathbf{x}_\alpha) \prod_{j \in N(\alpha)\setminus i} n_{j\to\alpha}(\mathbf{x}_\alpha)\right)^{1/\epsilon}\right)^{\epsilon}$$

$$n_{i\to\alpha}(x_i) \propto \phi_i(x_i) \prod_{\beta \in N(i)\setminus\alpha} m_{\beta\to i}(x_i)$$

which is the sum-product algorithm for $\epsilon = 1$ and the max-product algorithm for $\epsilon = 0$.

When the factors corresponds to pairwise interactions $\alpha = (i,j)$ the messages of norm-product algorithm $m_{\alpha\to i}$ and $n_{i\to\alpha}$ can be written by the shorthand notation $m_{j\to i}$ and $n_{i\to j}$. The messages $m_{j\to i}$ of the norm-product algorithm in Fig. 1 depends on a single message $n_{j\to i}$ and whenever $c_{i\alpha} = 0$ the message $n_{j\to i}$ depends only on the messages $m_{k\to j}$ for every $\{k,j\} \in N(j)$, which we abbreviate by $k \in N(j)$. Substituting the value of $n_{j\to i}$ into $m_{j\to i}$ we obtain the pairwise norm-product, whose update rule consists only of the messages $m_{k\to j}$. When $\epsilon = 1$ the pairwise norm-product algorithm with $c_{i\alpha} = 0$ takes the form

$$m_{j\to i}^{1/c_{ij}}(x_i) \propto \sum_{x_j} \psi_{ij}^{1/c_{ij}}(x_i,x_j) \frac{\phi_j^{1/\hat{c}_j}(x_j) \prod_{k \in N(j)} m_{k\to j}^{1/\hat{c}_j}(x_j)}{m_{i\to j}^{1/c_{ij}}(x_j)}.$$

The sum-TRBP [62] is a special case. The sum-TRBP sets c_{ij} as the relative number of spanning trees of the graph which include the edge (i,j), and sets $c_i = 1 - \sum_{j \in N(i)} c_{ij}$. As a result $\hat{c}_i = 1$ and by substitution $M_{ij}(x_i) \overset{def}{=} m_{j\to i}^{1/c_{ij}}(x_i)$ we obtain the sum-TRBP update rule as originally introduced in ([62], eqn. 39):

$$M_{ij}(x_i) \propto \sum_{x_j} \psi_{ij}^{1/c_{ij}}(x_i,x_j) \frac{\phi_j(x_j) \prod_{k \in N(j)} M_{jk}^{c_{jk}}(x_j)}{M_{ji}(x_j)}.$$

When $\epsilon = 0$ the pairwise norm-product algorithm with $c_{i\alpha} = 0$ takes the form

$$m_{j\to i}(x_i) \propto \max_{x_j} \psi_{ij}(x_i,x_j) \frac{\phi_j^{c_{ij}/\hat{c}_j}(x_j) \prod_{k \in N(j)} m_{k\to j}^{c_{ij}/\hat{c}_j}(x_j)}{m_{i\to j}(x_j)}.$$

The max-TRBP [63] and NMPLP [18] are special cases, derived as follows: With max-TRBP, we have c_{ij} and c_i defined by the tree-reweighted setting which results in $\hat{c}_i = 1$, and the Max-TRBP ([63], eqn. 50) follows from the substitution $M_{ij}(x_i) \overset{def}{=} m_{j\to i}^{1/c_{ij}}(x_i)$. The NMPLP is another recent max-product-like algorithm where messages $\gamma_{ji}(x_i)$ are defined as follows:

$$\gamma_{ji}(x_i) = \max_{\mathbf{x}_j} \left\{ \theta_{ij}(x_i,x_j) - \gamma_{ij}(x_j) + w_j \sum_{k \in N(j)} \gamma_{kj}(x_j) \right\}$$

Algorithm 3 (Sum-Product Belief Propagation type): We are given nonnegative local evidence $\phi_i(x_i)$, and nonnegative arrays $\psi_\alpha(\mathbf{x}_\alpha)$, where $\alpha \subset \{1,...,n\}$. Let $\hat{c}_{i\alpha} = c_\alpha + c_{i\alpha}$ and $\hat{c}_i = c_i + \sum_{\alpha \in N(i)} c_\alpha$.

```
1) Set $n_{i\to\alpha}(\mathbf{x}_\alpha) = 1$ for all $i = 1,...,n$, $\alpha \in N(i)$ and $\mathbf{x}_\alpha$.
2) For $t = 1,2,...$
    a) For $i = 1,...n$ do:
```

$$\forall x_i \; \forall \alpha \in N(i) \quad m_{\alpha\to i}(x_i) = \left(\sum_{\mathbf{x}_\alpha \setminus x_i} \left(\psi_\alpha(\mathbf{x}_\alpha) \prod_{j \in N(\alpha)\setminus i} n_{j\to\alpha}(\mathbf{x}_\alpha) \right)^{1/\hat{c}_{i\alpha}} \right)^{\hat{c}_{i\alpha}}$$

$$\forall \alpha \in N(i) \; \forall \mathbf{x}_\alpha \quad n_{i\to\alpha}(\mathbf{x}_\alpha) \propto \left(\frac{\phi_i^{1/\hat{c}_i}(x_i) \prod_{\beta \in N(i)} m_{\beta\to i}^{1/\hat{c}_i}(x_i)}{m_{\alpha\to i}^{1/\hat{c}_{i\alpha}}(x_i)} \right)^{c_\alpha} \left(\psi_\alpha(\mathbf{x}_\alpha) \prod_{j \in N(\alpha)\setminus i} n_{j\to\alpha}(\mathbf{x}_\alpha) \right)^{-c_{i\alpha}/\hat{c}_{i\alpha}}$$

Fig. 2. Sum-product belief propagation type algorithm, attained from the norm-product belief propagation when $\epsilon = 1$, where the messages $m_{\alpha\to i}(x_i)$ are computed with the $L_{1/\hat{c}_{i\alpha}}$ norm. For $c_\alpha = 1, c_i = 1 - d_i, c_{i\alpha} = 0$ it reduces to the sum-product belief propagation algorithms, and whenever c_α is the weighted number of spanning trees through edge α, and $c_i = 1 - \sum_{\alpha \in N(i)} c_\alpha$ and $c_{i\alpha} = 0$ it reduces to sum-TRBP. If $c_\alpha > 0, c_i, c_{i\alpha} \geq 0$ it reduces to the convex-sum-product algorithm, which is guaranteed to reach the global optimum of the convex free energy.

where $w_j = 2/(d_j + 1)$. The pairwise norm-product message $m_{j\to i}(x_i)$ with the setting $c_j = (1 - d_j)/2$ and $c_{ij} = 1$ for every (i,j) gives rise to $c_{ij}/\hat{c}_j = 2/(d_j + 1)$. Thus with the substitution $\gamma_{ji}(x_i) \overset{def}{=} \ln m_{j\to i}(x_i)$ and unit local potentials $(\phi_i(x_i) = 1)$ we obtain the NMPLP message above.

The result of having the BP, TRBP and NMPLP algorithms arise as special cases of the norm-product algorithm underscores the generality of our derivation. However, the more interesting potential in the norm-product algorithm is the emergence of *new* message-passing schemes which are guaranteed to converge (unlike the BP and TRBP algorithms) corresponding to the setting of $\tilde{H}$ as a concave function $(c_\alpha > 0, c_i, c_{i\alpha} \geq 0)$. Three classes of algorithms emerge:

- The *convex-sum-product* corresponding to the setting $\epsilon = 1$ in the norm-product algorithm. The convex-sum-product is guaranteed to converge to the global optimum of the primal function eqn. 7. This includes the tree-reweighted free-energy in particular and other settings of convex-free-energy which are detailed in Appendix D.
- The *approximate LP-relaxation* corresponding to the setting $\epsilon \to 0$ (but $\epsilon > 0$) in the norm-product algorithm. It provides an approximate solution to the LP-relaxation whose distance from the true solution is governed by an upper-bound we derive. The approximate LP-relaxation is guaranteed to converge to the global optimum of the primal function eqn. 7.
- The *convex-max-product* corresponding to the setting $\epsilon = 0$ in the norm-product algorithm. Unlike the max-product, the convex-max-product is convergence guaranteed. However, there is no guarantee that the recovered solution corresponds to the desired LP-relaxation solution. The advantage of convex-max-product is efficiency (introduced by L_∞ instead of $L_{1/\epsilon}$) and very good empirical performance. In fact, the convex-max-product is a convergent form of max-product.

These message-passing algorithms, which are collectively referred to as *convex-BP* algorithms, are discussed in the next section.

IV. The Convex Belief Propagation Algorithms

Eqn. 7 represents the free-energy approximation when $\epsilon = 1$, the LP relaxation when $\epsilon = 0$, and a perturbation of the LP-relaxation for MAP estimation when $\epsilon \to 0$. When the entropy approximation term $\tilde{H}$ is the Bethe approximation (setting $c_\alpha = 1, c_i = 1 - d_i, c_{i\alpha} = 0$ in eqn. 4) the sum-product $(\epsilon = 1)$ and max-product $(\epsilon = 0)$ arise as special cases of the norm-product algorithm. Since in both cases the free-energy approximation is non-convex (for factor graphs with cycles) the convergence guarantees of those algorithms are weak. For the sum-product we have the guarantee that *if* the algorithm convergence then it will reach a stationary point of the free-energy approximation (see Claim 2 and [69]). With the max-product we have weaker guarantees (Claim 2 does not apply because f_ϵ is not strictly convex when $\epsilon = 0$) where specifically, even if the algorithm does converge the marginal consistency constraints might not be satisfied.

We focus now on the family of convex-free-energies which arise with the setting $c_\alpha > 0, c_i, c_{i\alpha} \geq 0$. The convex-sum-product arises from the setting $\epsilon = 1$ is described next.

A. Convex-sum-product Algorithm

As a free-energy approximation $(\epsilon = 1)$, eqn. 7 is strictly convex and, in turn, the norm-product algorithm is guaranteed to converge to the global optimum. We refer to the specialization of the norm-product algorithm with $c_\alpha > 0, c_i, c_{i\alpha} \geq 0$ and $\epsilon = 1$ as *convex-sum-product* summarized in Fig. 2.

The beliefs $b_i(x_i)$, which are the approximations to the marginal probability $p(x_i)$, and the joint beliefs $b_\alpha(\mathbf{x}_\alpha)$, which are the approximation to the marginal probability $p(\mathbf{x}_\alpha)$, are computed from:

$$b_i(x_i) \propto \left(\phi_i(x_i) \prod_{\alpha \in N(i)} m_{\alpha \to i}(x_i)\right)^{1/\hat{c}_i},$$

$$b_\alpha(\mathbf{x}_\alpha) \propto \left(\psi_\alpha(\mathbf{x}_\alpha) \prod_{j \in N(\alpha)} n_{j \to \alpha}(\mathbf{x}_\alpha)\right)^{1/c_\alpha}.$$

Note that the algorithm has a much simpler form if $c_{i\alpha} = 0$. The message $n_{i\to\alpha}(\mathbf{x}_\alpha)$ depends only on x_i and becomes:

$$n_{i\to\alpha}(x_i) \propto \frac{\left(\phi_i(x_i) \prod_{\beta \in N(i)} m_{\beta \to i}(x_i)\right)^{c_\alpha/\hat{c}_i}}{m_{\alpha\to i}(x_i)}. \qquad (14)$$

The convex-sum-product is globally convergent for any concave setting of the entropy approximation $\tilde{H}$, i.e., when $c_\alpha > 0, c_i, c_{i\alpha} \geq 0$. In particular, when the underlying factor-graph arises from a graph, i.e., the local interaction forms pairwise relations only, there is a setting that corresponds to TRW free-energy as described in Appendix D. We also describe there additional parameter settings corresponding to other heuristic convex approximations of the entropy term $\tilde{H}$.

We describe next the use of the norm-product algorithm as an approximation to the LP-relaxation for the MAP problem by taking $\epsilon \to 0$.

B. LP-relaxation Bounds

For $\epsilon > 0$, let the global optimum of eqn. 7 (with concave $\tilde{H}$) denoted by $\mathbf{b}_\epsilon$ and let the solution of the LP relaxation eqn. 6 denoted by $\mathbf{b}^*$. Let $\boldsymbol{\theta}$ stand for the concatenated functions $\theta_i(x_i)$ and $\theta_\alpha(\mathbf{x}_\alpha)$, i.e., $\boldsymbol{\theta}^\top \mathbf{b} = \sum_{i,x_i} \theta_i(x_i) b_i(x_i) + \sum_{\alpha,\mathbf{x}_\alpha} \theta_\alpha(\mathbf{x}_\alpha) b_\alpha(\mathbf{x}_\alpha)$. We wish to upper-bound the difference $\boldsymbol{\theta}^\top \mathbf{b}_\epsilon - \boldsymbol{\theta}^\top \mathbf{b}^* \leq \delta$ where δ is a function of ϵ, c_α, c_i and $c_{i\alpha}$, described below:

Proposition 1: Let $c_\alpha > 0, c_i, c_{i\alpha} \geq 0$ describe a convex-free-energy eqn. 7. Let n_i stand for the cardinality of x_i and $n_\alpha = \prod_{i \in N(\alpha)} n_i$ be the cardinality of $\mathbf{x}_\alpha$. Then,

$$0 \leq \boldsymbol{\theta}^\top \mathbf{b}_\epsilon - \boldsymbol{\theta}^\top \mathbf{b}^* \leq \delta,$$

where

$$\delta = \epsilon \left(\sum_\alpha c_\alpha \ln n_\alpha + \sum_i c_i \ln n_i + \sum_i \sum_{\alpha \in N(i)} c_{i\alpha} \ln \frac{n_\alpha}{n_i}\right).$$

Proof: The sets of beliefs $\mathbf{b}^*, \mathbf{b}_\epsilon$ are both in the local polytope $\mathbb{L}(G)$ whereas the beliefs $\mathbf{b}^*$ are the optimal ones with respect to the original linear program eqn. 6, therefore $\boldsymbol{\theta}^\top \mathbf{b}^* \leq \boldsymbol{\theta}^\top \mathbf{b}_\epsilon$. On the other hand the beliefs $\mathbf{b}_\epsilon$ are optimal for the perturbed program eqn. 7, hence $\boldsymbol{\theta}^\top \mathbf{b}_\epsilon \leq \boldsymbol{\theta}^\top \mathbf{b}^* + \epsilon(\tilde{H}(\mathbf{b}_\epsilon) - \tilde{H}(\mathbf{b}^*))$ where $\tilde{H}(\mathbf{b})$ is described in eqn. 4.

Using Jensen's inequality we obtain:

$$H(\mathbf{b}_i) = \sum_{x_i} b_i(x_i) \ln \frac{1}{b_i(x_i)} \leq \ln \sum_{x_i} \frac{b_i(x_i)}{b_i(x_i)} = \ln n_i,$$

and likewise $H(\mathbf{b}_\alpha) \leq \ln n_\alpha$. Substituting in eqn. 4 and noting that $\tilde{H}(\mathbf{b}^*) \geq 0$ we obtain:

$$\tilde{H}(\mathbf{b}_\epsilon) - \tilde{H}(\mathbf{b}^*) \leq \sum_\alpha c_\alpha \ln n_\alpha + \sum_i c_i \ln n_i + \sum_{i,\alpha} c_{i\alpha} \ln \frac{n_\alpha}{n_i}.$$

□

As a result, in the ideal world, one could generate the solution $\mathbf{b}_\epsilon$ arbitrarily close to the relaxed LP solution $\mathbf{b}^*$. There are, however, numerical accuracy limitations which in practice limit the size of $\epsilon \geq \epsilon_0 > 0$. The assumption in Proposition 1 is that the output $\mathbf{b}_\epsilon^{n.p.}$ of the norm-product algorithm, as defined in eqns. 12,13, is equal to $\mathbf{b}_\epsilon$ the solution to the ϵ-perturbed LP-relaxation eqn. 7. This is indeed true when $\epsilon > 0$ but not when $\epsilon = 0$. As we shall see in more details in the next section, the norm-product algorithm is guaranteed to converge when $\epsilon = 0$ but not necessarily to the minimal primal value. Therefore, from a numerical perspective there exists ϵ_0 such that when $\epsilon < \epsilon_0$ the underlying assumption $\mathbf{b}_\epsilon^{n.p.} = \mathbf{b}_\epsilon$ ceases to hold. Moreover, the value of ϵ_0 depends on the graph structure and the potential functions ψ_α and therefore is unlikely to have a simple and useful form.

C. Convex-max-product Algorithm

We saw that for the setting of $\epsilon = 0$ and when $\tilde{H}$ equals the Bethe entropy approximation then the norm-product becomes the max-product algorithm. We now explore the convex-free-energy setting $c_\alpha > 0, c_i, c_{i\alpha} \geq 0$ while $\epsilon = 0$ and refer to the resulting family of algorithms as *convex-max-product* summarized in Fig. 3.

Note that when $c_{i\alpha} = 0$ we obtain a much simpler form of the algorithm where the message $n_{i\to\alpha}(\mathbf{x}_\alpha)$ depends only on x_i described in eqn. 14:

Algorithm 5 (Convex-Max-Product when $c_{i\alpha} = 0$): Repeat until convergence:

1) For $i = 1, ...n$ and for all $\alpha \in N(i)$ do:

$$m_{\alpha\to i}(x_i) = \max_{\mathbf{x}_\alpha \setminus x_i} \left\{\psi_\alpha(\mathbf{x}_\alpha) \prod_{j \in N(\alpha)\setminus i} n_{j\to\alpha}(x_j)\right\}$$

$$n_{i\to\alpha}(x_i) \propto \frac{\left(\phi_i(x_i) \prod_{\beta \in N(i)} m_{\beta\to i}(x_i)\right)^{c_\alpha/\hat{c}_i}}{m_{\alpha\to i}(x_i)}$$

The desired output vector $b_i(x_i)$ is recovered from computing the vector $\phi_i^{1/\hat{c}_i}(x_i) \prod_{\alpha \in N(i)} m_{\alpha\to i}^{1/\hat{c}_i}(x_i)$ as follows. If there are no ties, $b_i(x_i)$ is determined by setting the highest value to 1 and all remaining entries to 0. If the highest value of the vector is shared among $r_i > 1$ entries, i.e., there exist ties, then those entries receive the value $1/r_i$. If there are no ties, i.e., $r_i = 1$ for $i = 1, ..., n$, then the result is the MAP solution.

The setting $\epsilon = 0$ raises two issues (i) if the algorithm converges, can one obtain from them the optimal LP-relaxation solution?, and (ii) is there a convergence guarantee of the convex-max-product family? The answer to the first question is generally negative. In a nutshell, the primal function $f_{\epsilon=0}$ is convex but no longer strictly convex and therefore the dual

Algorithm 4 (Max-Product Belief Propagation type): We are given nonnegative local evidence $\phi_i(x_i)$, and nonnegative arrays $\psi_\alpha(\mathbf{x}_\alpha)$, where $\alpha \subset \{1,...,n\}$. Let $\hat{c}_{i\alpha} = c_\alpha + c_{i\alpha}$ and $\hat{c}_i = c_i + \sum_{\alpha \in N(i)} c_\alpha$.

```
1) Set $n_{i\to\alpha}(\mathbf{x}_\alpha) = 1$ for all $i = 1,...,n$, $\alpha \in N(i)$ and $\mathbf{x}_\alpha$.
2) For $t = 1,2,...$
    a) For $i = 1,...n$ do:
```

$$\forall x_i \; \forall \alpha \in N(i) \quad m_{\alpha\to i}(x_i) = \max_{\mathbf{x}_\alpha \setminus x_i} \left\{ \psi_\alpha(\mathbf{x}_\alpha) \prod_{j \in N(\alpha)\setminus i} n_{j\to\alpha}(\mathbf{x}_\alpha) \right\}$$

$$\forall \alpha \in N(i) \; \forall \mathbf{x}_\alpha \quad n_{i\to\alpha}(\mathbf{x}_\alpha) \propto \left(\frac{\phi_i^{1/\hat{c}_i}(x_i) \prod_{\beta \in N(i)} m_{\beta\to i}^{1/\hat{c}_i}(x_i)}{m_{\alpha\to i}^{1/\hat{c}_{i\alpha}}(x_i)} \right)^{c_\alpha} \left(\psi_\alpha(\mathbf{x}_\alpha) \prod_{j \in N(\alpha)\setminus i} n_{j\to\alpha}(\mathbf{x}_\alpha) \right)^{-c_{i\alpha}/\hat{c}_{i\alpha}}$$

Fig. 3. Max-product belief propagation type algorithm, attained from the norm-product belief propagation when $\epsilon = 0$, where the messages $m_{\alpha\to i}(x_i)$ are computed with the L_∞ norm. For $c_\alpha = 1, c_i = 1 - d_i, c_{i\alpha} = 0$ it reduces to the max-product belief propagation algorithms. Whenever c_α is the weighted number of spanning trees through edge α, and $c_i = 1 - \sum_{\alpha \in N(i)} c_\alpha$ and $c_{i\alpha} = 0$ it reduces to max-TRBP. For $c_\alpha = 1, c_i = (1-d_i)/2, c_{i\alpha} = 0$ it reduces to the NMPLP algorithm. If $c_\alpha > 0, c_i, c_{i\alpha} \geq 0$ it reduces to the convex-max-product algorithm, which is a convergent max-product type algorithm for LP-relaxations.

function is no longer differentiable. A dual ascent approach on a non-differentiable dual function can get stuck at "corners". The implication of getting stuck at a corner of the energy landscape is that the recovered primal solution $\mathbf{b}_{\epsilon=0}$ might not correspond to the lowest primal energy and furthermore might not satisfy the marginal consistency constraints. More details can be found in Appendix B-B.

We consider now the the second question of whether the dual ascent creates a converging sequence? The answer is positive, i.e., the convex-max-product algorithm is convergent (unlike max-product on general graphs).

Theorem 1 (Convergence, Convex-max-product): The norm-product algorithm with the parameter setting of $\epsilon = 0$ and $c_\alpha > 0, c_i, c_{i\alpha} \geq 0$ is convergent.

Proof: Let $q_\epsilon(\boldsymbol{\lambda}_1,...,\boldsymbol{\lambda}_n)$ represent the conjugate dual eqn. 21:

$$q_\epsilon(\boldsymbol{\lambda}_1,...,\boldsymbol{\lambda}_n) = -f_\epsilon^*(-\sum_i \boldsymbol{\lambda}_i) - \sum_{i=1}^n h_{\epsilon,i}^*(\boldsymbol{\lambda}_i),$$

and let $q_0(\boldsymbol{\lambda}_1,...,\boldsymbol{\lambda}_n)$ be the limit of q_ϵ as $\epsilon \to 0$. The explicit form of the conjugate duals f_ϵ^* and $h_{\epsilon,i}^*$ are:

$$f_\epsilon^*(\boldsymbol{\lambda}) = \sum_\alpha \ln \|\psi_\alpha(\mathbf{x}_\alpha)\exp(\lambda_\alpha(\mathbf{x}_\alpha))\|_{1/\epsilon c_\alpha} \quad (15)$$

$$h_{\epsilon,i}^*(\boldsymbol{\lambda}) = \ln \left\| \phi_i(x_i) \prod_{\alpha \in N(i)} \|_{\mathbf{x}_\alpha \setminus x_i} \exp(\lambda_\alpha(\mathbf{x}_\alpha))\|_{1/\epsilon c_{i\alpha}} \right\|_{1/\epsilon c_i} \quad (16)$$

where $\|_{\mathbf{x}_\alpha \setminus x_i} z(\mathbf{x}_\alpha)\|_p^p = \sum_{\mathbf{x}_\alpha \setminus x_i} |z_\alpha(\mathbf{x}_\alpha)|^p$. The functions $f_0^* \overset{def}{=} f_{\epsilon\to 0}^*$ and $h_{0,i}^* \overset{def}{=} h_{\epsilon\to 0,i}^*$ are well defined and thus,

$$q_0(\lambda_1,...,\lambda_n) = -f_0^*(-\sum_i \boldsymbol{\lambda}_i) - \sum_{i=1}^n h_{0,i}^*(\boldsymbol{\lambda}_i),$$

is well defined as well. By definition of the block ascent scheme, let $\boldsymbol{\lambda}_{\epsilon,i} \in \text{argmax}_{\boldsymbol{\lambda}_i} q_\epsilon(\boldsymbol{\lambda}_1,...,\boldsymbol{\lambda}_n)$. We note that $\boldsymbol{\lambda}_{0,i} = \lim_{\epsilon\to 0} \boldsymbol{\lambda}_{\epsilon,i}$ is well defined because ϵ appears as a norm in the definition of the message $n_{i\to\alpha}$.

We use the shorthand $q_\epsilon(\boldsymbol{\lambda}_{\epsilon,i})$ instead of $q_\epsilon(\boldsymbol{\lambda}_1,...,\boldsymbol{\lambda}_{i-1},\boldsymbol{\lambda}_{\epsilon,i},\boldsymbol{\lambda}_{i+1},...,\boldsymbol{\lambda}_n)$. We wish to show that $\boldsymbol{\lambda}_{0,i} \in \text{argmax}_{\boldsymbol{\lambda}_i} q_0(\boldsymbol{\lambda}_1,...,\boldsymbol{\lambda}_n)$.

Assume to the contrary that $\boldsymbol{\lambda}_{0,i} \notin \text{argmax}_{\boldsymbol{\lambda}_i} q_0(\cdot)$ and let instead $\hat{\boldsymbol{\lambda}}_{0,i} \in \text{argmax}_{\boldsymbol{\lambda}_i} q_0(\cdot)$, thus making $q_0(\hat{\boldsymbol{\lambda}}_{0,i}) > q_0(\boldsymbol{\lambda}_{0,i})$. Since $q_0 = \lim_{\epsilon\to 0} q_\epsilon$, there exists ϵ_0 such that for all $\epsilon \leq \epsilon_0$ we have $q_\epsilon(\hat{\boldsymbol{\lambda}}_{0,i}) > q_0(\boldsymbol{\lambda}_{0,i})$ as well. Likewise, using the limit argument on the right-hand side, $q_\epsilon(\hat{\boldsymbol{\lambda}}_{0,i}) > q_\epsilon(\boldsymbol{\lambda}_{0,i})$. Finally, since $\boldsymbol{\lambda}_{0,i} = \lim_{\epsilon\to 0} \boldsymbol{\lambda}_{\epsilon,i}$, and q_ϵ is continuous, we have $q_\epsilon(\hat{\boldsymbol{\lambda}}_{0,i}) > q_\epsilon(\boldsymbol{\lambda}_{\epsilon,i})$ which contradicts the fact that $\boldsymbol{\lambda}_{\epsilon,i} \in \text{argmax}_{\boldsymbol{\lambda}_i} q_\epsilon(\cdot)$. □

We conclude that the convex-max-product, unlike max-product, is convergence guaranteed, since it iteratively improves the dual objective which is bounded by the primal objective. The convex max-product is guaranteed to recover the MAP assignment if its beliefs are integral. However, in many cases we can use the rounding scheme for the max-product type algorithms which guarantees the MAP if the beliefs recovered from the messages are without ties [65].

V. Experiments

In our experiments we first evaluated the quality of the max-product type algorithms for solving a linear program with pairwise interactions and binary variables

$$\min_{\mathbf{b}_i, \mathbf{b}_{i,j} \in \mathbb{L}(G)} \sum_{i, x_i \in \{0,1\}} \theta_i(x_i) b_i(x_i) + \sum_{(i,j) \in E, x_i, x_j \in \{0,1\}} \theta_{i,j}(x_i,x_j) b_{i,j}(x_i,x_j)$$

The max-product type algorithms differ from each other by their approximated entropy coefficients $c_\alpha, c_i, c_{i\alpha}$, but since the linear program has no entropy terms, all these algorithms aim at producing the same result. We distinguish between three families of max-product type algorithms:

- The first family corresponds to non-concave entropy approximation, such as the Bethe free energy whose coefficients $c_\alpha = 0, c_i = 1 - d_i$ and $c_{i\alpha} = 0$ produce the max-product algorithm. These algorithms are not guaranteed to converge and even if they converge there are no guarantees on their solution.

- The second family corresponds to concave entropy approximations with positive c_α, negative c_i and $c_{i\alpha} = 0$. The notable member of this family is the max-TRBP algorithm [63], whose c_α is the weighted number of spanning trees which pass through the edge α and $c_i = 1 - \sum_{\alpha \in N(i)} c_\alpha$. These max-product type algorithms are not guaranteed to converge, but whenever they converge one can extract an optimal solution for a pairwise linear program with binary variables, cf. [31] theorem 4 and [38] corollary 2.
- The third family corresponds to concave entropy approximation with $c_\alpha, c_i, c_{i\alpha} \geq 0$. These convex-max-product algorithms are guaranteed to converge to the global optimum for a pairwise linear program with binary variables, cf. [38] corollary 2 and [18] proposition 3.

We used the implementation of the max-product type algorithm described in Algorithm 4, while each algorithm differs in its appropriate $c_\alpha, c_i, c_{i\alpha}$. To evaluate the performance of the algorithms we generated 100 samples of 10×10 grids, where θ_i and $\theta_{i,j}$ were sampled from zero mean Gaussians with standard deviation of one. We set the local evidence according to $\theta_i(x_i) = \theta_i(-1)^{x_i}$, and for the pairwise interactions $\theta_{i,j}(x_i, x_j)$ we set the value $\theta_{i,j}$ on their diagonal and $-\theta_{i,j}$ on their off-diagonal.

First we investigated the convergence properties of three representatives of the max-product families described above: The max-product algorithm, the max-TRBP described in [63], and the convex max-product with the same tree-reweighted free energy, represented by $c_\alpha, c_i, c_{i\alpha} \geq 0$ as described in Appendix D. The convergence criterion for the max-product and max-TRBP algorithms was measured with respect to change in their messages, whereas the convergence criterion for the convex-max-product was measured with respect to change in its dual function. The max-product algorithm converged for 25% of the runs, the max-TRBP converged for 90% of the runs, and as expected from Theorem 1 the convex-max-product always converged. However, the convex max-product was slower than max-TRBP, while we measured the primal values obtained by both algorithms during their runs. Over the runs the max-TRBP converged in average number of 430 iterations compared to an average of 6400 of the convex-max-product with tree-reweighted parameters.

Next we compared the run-time of three representatives of the converging max-product: The convex-max-product with tree-reweighted free energy, the NMPLP of [18] and the convex-max-product with $c_\alpha = 1, c_i = 0, c_{i\alpha} = 0$ which was referred as "trivial convex-max-product" by [65]. We measured their convergence with respect to the change in their dual objective: The NMPLP converged in average number of 200 iterations, the trivial convex-max-product converged in average of 260 iterations, and the convex-max-product with tree-reweighted free energy converged in average of 6400 iterations.

To conclude, for linear programs with pairwise interactions and binary variables the convex-max-product algorithms improve upon previous max-product type algorithms: They are guaranteed to converge to the global optimum. However the convex-max-product algorithms differ from each other in their memory requirements and run-time. Among those algorithms, the ones with $c_{i\alpha} = 0$ requires less memory, as their messages $n_{i\to\alpha}$ depend only on x_i, and have a faster run-time.

The norm-product family of algorithms can also solve linear program using the perturbation method for a small value of ϵ, as described in Proposition 1. However the convex-max-product algorithms are computationally more efficient, and guaranteed converge to the global optimum of linear program with pairwise interactions and binary variables. Therefore we evaluate the convex-norm-product type algorithms over linear programs with non-binary variables

$$\min_{b_i, b_{i,j} \in \mathbb{L}(G)} \sum_{i, x_i \in \{1,2,3\}} \theta_i(x_i) b_i(x_i) + \sum_{(i,j) \in E, x_i, x_j \in \{1,2,3\}} \theta_{i,j}(x_i, x_j) b_{i,j}(x_i, x_j)$$

For these programs the convex-norm-product algorithms are guaranteed to converge to the global optimum, whereas the convex-max-product can converge to non-optimal stationary point. To evaluate the performance of the convex-norm-product we generated 100 samples of 10×10 grid where $\theta_i(x_i)$ and $\theta_{i,j}$ were sampled from zero mean Gaussians with standard deviation of one, and $\theta_{i,j}(x_i, x_j)$ were given the value $\theta_{i,j}$ on their diagonal and $-\theta_{i,j}$ on their off-diagonal.

We measured how often the convex-max-product algorithm converges to non-optimal stationary points, comparing to the convex-norm-product which always achieves its optimum as described in Claim 8. To indicate these events we compared the dual value of the linear program, which is evaluated by the convex-max-product stationary messages and by the convex-norm-product messages, setting $\epsilon = 0.001$ and $c_\alpha = 1, c_i = 0, c_{i\alpha} = 0$. For 60% of the runs, the dual values attained by the convex-max-product and the convex-norm-product were 0.01 close to each other, indicating both algorithms reached the maximal dual value. On the other hand, for 25% of the runs the dual value of the linear program attained by the convex-max-product messages was 0.1 lower than the one attained by the convex-norm-product messages, indicating the convex-max-product reached a non-maximal dual value. This fact has important practical implications: Only from the dual optimal solution one can recover the optimal beliefs that solve the primal linear program, while non-optimal dual messages always relate to non-consistent beliefs. In particular for the 25% of the runs the convex-max-product did not produce beliefs which agree on their marginal probabilities, whereas the convex-norm-product always recover beliefs which satisfy the primal linear program constraints.

In our experiments we also evaluated the sum-product type algorithms for approximating the marginal probabilities of distribution $p(\mathbf{x})$ of the form

$$p(\mathbf{x}) \propto \exp\left(\sum_{i, x_i} \theta_i(x_i) + \sum_{\alpha, \mathbf{x}_\alpha} \theta_\alpha(\mathbf{x}_\alpha)\right)$$

The variational framework for approximating marginal probabilities, described in Section II-A, suggests that the approximated entropy term affects the quality of the approximated marginal probabilities. Although we do not have a theoretical guarantee for setting the best approximation, in these experiments we show how the different approximations behave in practice. We consider two types of free energy approximations:

- Non-convex free energy approximations, represented by the Bethe approximation which corresponds to $c_\alpha = 1, c_i = 1 - d_i, c_{i\alpha} = 0$. The sum-product algorithm aims at finding a local minimum for the Bethe free energy approximation, but it is not guaranteed to converge. In cases where it does not converge we used the double loop algorithm [22] in libDAI [40], which is guaranteed to converge to a stationary point of the Bethe free energy.
- Free energy approximations which are convex in the intersection of the marginalization constraints. These approximations are appealing since their stationary points are their global minimum. We address the tree-reweighted free energy approximations whose $c_\alpha, c_i, c_{i\alpha}$ correspond to spanning trees in the graph, and also to L_2 convex free energy approximation heuristic described in Appendix D. We note that whenever $c_\alpha, c_i, c_{i\alpha} \geq 0$ the corresponding convex-sum-product algorithms are guaranteed to converge to the global optimum.

We used the implementation of the sum-product type algorithm described in Algorithm 3, while each algorithm differs in its appropriate $c_\alpha, c_i, c_{i\alpha}$. Following [62] We generated 100 samples of 10×10 grids with binary variables $x_i \in \{0, 1\}$, where θ_i were uniformly chosen from the interval $[-0.05, 0.05]$, and $\theta_{i,j}$ were either chosen uniformly from the *attractive* interval $[0, \omega]$ or the *mixed* interval $[-\omega, \omega]$. We ran the simulations with edge strength ω ranging from 0 to 2. We set the local evidence to $\theta_i(x_i) = \theta_i(-1)^{x_i}$, and for the pairwise interactions $\theta_{i,j}(x_i, x_j)$ we set the value $\theta_{i,j}$ on their diagonal and $-\theta_{i,j}$ on their off-diagonal.

We compared to true marginal probabilities with the approximated marginal probabilities recovered from the Bethe free energy approximation, tree-rewighted free energy approximation, and the L_2 convex-free-energy heuristic. Fig. 4 shows the average L_1 error in the marginal probabilities $\frac{1}{100} \sum_i |p^{(alg)}(x_i = 1) - p^{(true)}(x_i = 1)|$.

We conclude from this experiment that the convex approximations are better than the Bethe approximation for the attractive settings, when $\theta_{ij} \geq 0$. However, the Bethe approximation is slightly better in the mixed settings for $\omega < 1$ and considerably worse for $\omega > 1$. Moreover, in the mixed settings the sum-product did not converge for $\omega > 1$ and we used the double-loop algorithm instead which is computationally more expensive. We also conclude that the L_2 convex free energy settings produce comparable results to tree-rewiehted free energy for grids.

We also compared the tree-reweighted and L_2 convex free energy approximated marginal probabilities on the complete graph, i.e. every two vertices are connected with an edge. We generated 100 samples of complete graphs with 10 vertices with binary variables, where θ_i were uniformly chosen from the interval $[-0.05, 0.05]$, and $\theta_{i,j}$ were chosen uniformly from the interval $[0, \omega]$, for ω ranging from 0 to 2. Fig. 5 shows the average L_1 error in marginal probability, suggesting that in the case of complete graph, whose structure is far from a tree, the L_2 convex approximation heuristic is better than the tree-reweighted approximation for marginal probabilities estimation.

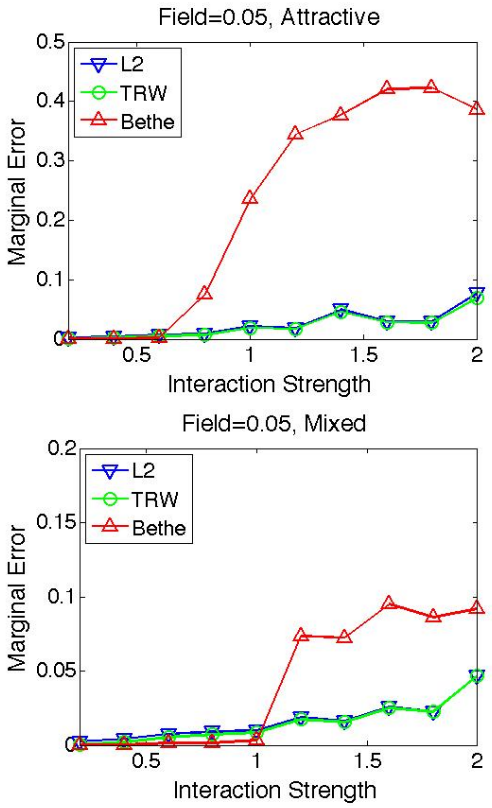

Fig. 4. *Comparison of error in marginal probabilities, estimated by Bethe free energy, tree-reweighted free energy and L_2 convex free energy described in Appendix D. We computed the Bethe approximation by applying the sum-product when converged, and the double-loop algorithm otherwise. The other free energy approximations are convex and the convex-sum-product algorithm is guaranteed to converge to their optimum. The graphs present the average error over* 100 *random trials*

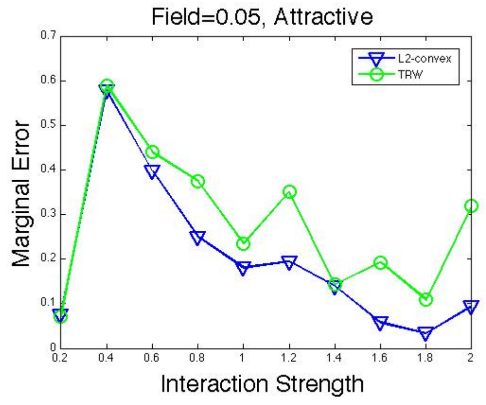

Fig. 5. *Comparison of error in marginal probabilities on a complete graph, estimated by tree-reweighted free energy approximation and L_2 convex free energy approximation. The graphs present the average error over* 100 *random trials*

Generally, the same convex free energy can be represented by different coefficients $c_\alpha, c_i, c_{i\alpha}$. In particular, the tree-reweighted free energy can be described by positive c_α, which correspond to the weighted number of spanning trees that go

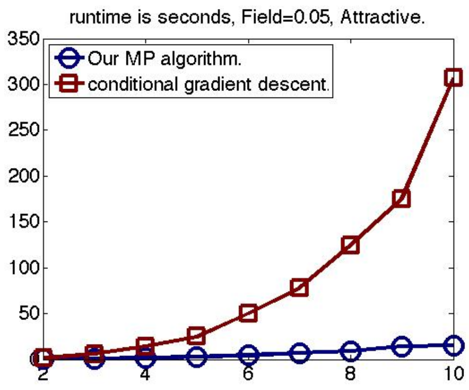

Fig. 6. *Run-time (in seconds) comparisons of convex-sum-product against a conditional gradient descent solver (running on convex-L_2 free energy). The algorithms were applied to $n \times n$ grids with $n = 2, 3..., 10$. Mean is shown for 10 random trials.*

through the edges α, and negative $c_i = 1 - \sum_{\alpha \in N(i)} c_\alpha$ and $c_{i\alpha} = 0$. However, the same tree-rewieghted free energy can be represented by $c_\alpha, c_i, c_{i\alpha} \geq 0$, as explained in Appendix D. These representations affect their corresponding sum-product type algorithms: The first representation corresponds to the sum-TRBP algorithm which is not guaranteed to converge, whereas the second representation corresponds to the convex-sum-product which is guaranteed to converge. However, the convex-sum-product was slower than sum-TRBP, while we measured the primal values obtained by both algorithms during their runs. Similar results were reported in [17].

Fig. 6 compares the running time of the convex-sum-product algorithm with a general convex solver performing conditional gradient descent on the primal energy function [4] which uses linear programming to find feasible search directions. We ran the algorithms on $n \times n$ grids where $n = 2, 3, ..., 10$. The stopping criteria for all algorithms was the same and based on a primal energy difference of 10^{-5}. For a 10×10 grid, for instance, the general convex solver was slower by a factor of 20 (e.g., 306 seconds compared to 15.2). For a 2×2 grid, on the other hand, convex-sum-product took 0.15 seconds compared 1.41 seconds for the general convex solver. We conclude that the sum-product type algorithms converge faster than a general convex solver, since they exploit the structure of the graph.

VI. Discussion

We have presented a single unified message-passing framework for approximate inference covering both marginal probabilities estimation and the MAP assignment problem through LP-relaxation. We took a general perspective on the existing BP and TRBP algorithms and noted that all are reductions from the basic optimization formula of $f + \sum_i h_i$ where the function f is an extended-valued, strictly convex but non-smooth and the functions h_i are extended-valued functions (not necessarily convex). We used tools from convex duality to present the "primal-dual ascent" algorithm which is an extension of the Bregman successive projection scheme. Most of the details of this part of the paper was pushed to Appendix B in order to reduce the overall technical load for the main-body presentation.

We then mapped the fractional-free-energy variational principal for approximate inference onto the optimization structure $f + \sum_i h_i$ and introduced the "norm-product" message-passing algorithm. Special cases of the norm-product include sum-product and max-product (BP algorithms), TRBP and NMPLP algorithms. When the fractional-free-energy is set to be convex (convex-free-energy) the norm-product is globally convergent for the estimation of marginal probabilities (the convex-sum-product branch corresponding to $\epsilon = 1$) and for approximating the LP-relaxation ($\epsilon \to 0$). We have also introduced another branch of the norm-product which arises as the "zero-temperature" of the convex-free-energy ($\epsilon = 0$) which we referred to as the convex-max-product. The convex-max-product is a convergent solver to the LP-relaxation (unlike max-product) but is not guaranteed to reach the global optimum.

As a general statement, the convex-free-energies provide a way for obtaining approximate inference over general graphs. There are two main issues in this regard: the first is how to obtain a guaranteed globally convergent message-passing algorithm for the general class of convex free energies, and secondly, how to tune the energy parameters $c_i, c_{i\alpha}, c_\alpha$ to a specific graph?

As for the first issue, we have provided a complete treatment which also encompasses the existing BP and TRBP algorithms (though they do not arise from a convex-free-energy but from a non-convex fractional-free-energy). As for the second issue, we provided a simple algorithm for converting the conventional TRW-free-energy settings to the convex-free-energy framework and have also proposed a heuristic principle where among all admissible parameters we choose the one most closest to the Bethe free energy (Appendix D). Empirical results show that for certain graphs, like a grid, we obtain very close marginal probability estimation results to those obtained by the TRW free energy. For complete graphs we obtain a very different free energy from TRW and superior accuracy of marginal probability estimation. The results suggest that our heuristic for setting up the convex free energy satisfies what we were after, i.e., to get approximations similar to BP but in guaranteed (globally) convergent framework.

In this work we limited the scope to factor graphs where the neighborhoods of every pair of factor nodes have at most a single intersection to give a clear description of the mathematical details presented in this work. However, the techniques presented here can also be used as a basis to a convex and non-convex generalized belief propagation [69]. Different algorithms were recently developed for tightening the LP-relaxation [51], [53], [32] using intersections of increasingly larger clusters in order to recover the MAP assignment. We believe similar techniques can be applied to convex free energies in order to tighten the bound on the log-partition function.

We did not discuss the parallel implementation of the norm-product algorithm, but as every message-passing algorithm it can be parallelized: One can distribute to the different parallel units an independent set of vertices, i.e. vertices which are not connected to each other in the graph. This mechanism preserves the convergence and optimal guarantees of the algorithm. The norm-product can also be made fully parallel, as it is a generalization of the belief propagation algorithm, but in this case convergence is no longer guaranteed. This can be fixed by methods described in [19].

The convergence rate and the complexity analysis of the norm-product algorithm were not addressed in this work. Since the convex norm-product algorithm performs a dual block ascent it has a linear convergence rate, whenever $\epsilon, c_\alpha, c_i > 0, c_{i\alpha} = 0$ (cf. [36] Theorem 5.1), i.e. it achieves a δ-optimal solution in $O(\log(1/\delta))$ steps. However, this notation does not capture the true complexity of the algorithm as $O(\log(1/\delta))$ depends on unknown constants which can be very large. For this purpose complexity bound were recently introduced, where it was proved that the dual gradient ascent attains linear complexity, (cf. [42] Theorem 2.1.13, [57] Theorem 5.1). Although the convex norm-product can be modified to achieve linear complexity its step size depends on ϵ, c_α, c_i and the modified algorithm is inefficient compared to the convex norm-product. We believe this due to the fact that the convex norm-product finds the optimal dual assignment $\boldsymbol{\lambda}_i$ in each step, unlike the gradient methods. Generally, a complexity bound for block coordinate ascent algorithms such as the convex norm-product is an open problem.

Future work is also required for obtaining a firmer theoretical understanding about how to set the concave entropy approximation, in order to guarantee a good approximation for the marginal probabilities. For example, how tight is the TRW-entropy bound, and whether one can find a family of trees which guarantees the best bound? Clearly, these theoretical guarantees must consider the potentials functions, since for every graph its TRW-entropy can be made arbitrary close to the true entropy for some potentials.

APPENDIX A
MATHEMATICAL BACKGROUND ON CONJUGATE DUALITY

We consider the n-dimensional Euclidean space R^n and denote vectors in bold face, e.g. $\mathbf{x} \in R^n$. We start with a brief review of basic concepts of sets. A set S is said to be *closed* if every of its limit points is contained the set. A set S is called *open* if its complement $R^n \setminus S$ is closed. The *interior* of a set S, denoted by $int(S)$, is the largest open set contained in S. The closure of a set, $cl(S)$, is the smallest closed set containing S. A point $\mathbf{x}$ is a *boundary* point of S if $\mathbf{x} \in cl(S)$ and $\mathbf{x} \notin int(S)$ or equivalently if every neighborhood of $\mathbf{x}$ contains at least one point of S and at least one point not of S. A set C is called *convex* if it contains the line-segment between any two points $\mathbf{x}$ and $\mathbf{y}$ in the set. That is, for every $0 \leq \lambda \leq 1$ the point $\lambda\mathbf{x} + (1-\lambda)\mathbf{y} \in C$.

For our purposes, since we deal with low-dimensional sets placed in higher-dimensional spaces, we use the concept of *relative interior* denoted by $ri(S)$ which, defined intuitively, contains all points which are not on the "edge" of the set, relative to the smallest affine subspace in which this set lies. For example, for a convex set C, $\mathbf{x} \in ri(C)$ if and only if $\forall \mathbf{y} \in C$ there exists $\mathbf{z} \in C$ and $0 < \lambda < 1$ such that $\mathbf{x} = \lambda\mathbf{z} + (1-\lambda)\mathbf{y}$.

The *graph* of a function $f(\mathbf{x})$ is the curve $\{(\mathbf{x}, f(\mathbf{x})) : \mathbf{x} \in R^n\}$, and define the *epigraph* of a function $f(\mathbf{x})$, denoted by $epi(f)$, as the set above its graph, namely $\{(\mathbf{x}, r) : \mathbf{x} \in R^n, r \geq f(\mathbf{x})\}$. A functions is called *closed* if its epigraph is a closed set. A function is said to be *convex* if its epigraph is a convex set. A function is called *strictly convex* if any line segment in its epigraph intersects with its relative interior. A twice differentiable function is convex if its matrix of second derivatives, called the *Hessian*, is positive semidefinite, and strictly convex if its Hessian is positive definite.

In this paper we work with functions that can take the value of infinity and as such are non-differentiable. Such functions are known as *extended-valued*:

Definition 2 (Extended-Valued, Proper): A function $f(\mathbf{x})$ is said to be extended real-valued if $-\infty \leq f(\mathbf{x}) \leq \infty$. The effective domain of $f(\mathbf{x})$ is denoted by $dom(f) = \{\mathbf{x} : f(\mathbf{x}) < \infty\}$. A function is said to be proper if $-\infty < f(\mathbf{x}) \leq \infty$, and it obtains at least one finite value.

Proper functions typically arise when constraints are embedded into finite valued functions. For example, the *indicator function* associated with a convex set C is defined by $\delta_C(\mathbf{x}) = 0$ when $\mathbf{x} \in C$ and $\delta_C(\mathbf{x}) = \infty$ otherwise. A possible use of the indicator function is to constrain a finite valued function $\hat{f}$ with the set convex set S to define a proper function $f = \hat{f} + \delta_S$. We define next the type of smoothness used throughout this paper:

Definition 3 (Essentially Smooth): Let f be a proper and convex function differentiable throughout the non-empty set $C = int(dom(f))$. Then f is called *essentially smooth* if $\lim_{k\to\infty} \|\nabla f(\mathbf{x}_k)\| = \infty$ whenever $\mathbf{x}_k$ is a sequence in C converging to a boundary point $\mathbf{x}$ in C.

Necessary and sufficient conditions for a function to be essentially smooth are described in the following theorem:

Theorem 2 (Legendre type): A closed and proper convex function $f(\mathbf{x})$ is essentially smooth if and only if it is differential in its interior $C = int(dom(f))$, i.e. $\partial f(\mathbf{x}) = \nabla f(\mathbf{x})$ for every $\mathbf{x} \in C$, while $\partial f(\mathbf{x}) = \emptyset$ when $\mathbf{x} \notin C$. If $f(\mathbf{x})$ is also strictly convex on C it is called a convex function of *Legendre type*, and its gradient mapping $\nabla f : C \to R^n$ is continuous and one-to-one, and $\nabla f^* = (\nabla f)^{-1}$.

Proof: [46], Theorem 26.1 and Theorem 26.5 □

The sets $\{\mathbf{x} : \mathbf{a}^\top\mathbf{x} \geq b\}$ and $\{\mathbf{x} : \mathbf{a}^\top\mathbf{x} \leq b\}$, are called the *closed half-spaces* associated with the hyperplane $\{\mathbf{x} : \mathbf{a}^\top\mathbf{x} = b\}$. We say that two sets C_1, C_2 are *separated by a hyperplane* if each set lies in a different closed halfspace associated with the hyperplane. If a vector $\bar{\mathbf{x}}$ is a boundary point of a set C, then a hyperplane that contains the singleton $\{\bar{\mathbf{x}}\}$ and one of its halfspaces contains C is said to be *supporting C at $\bar{\mathbf{x}}$*. In other words, a supporting hyperplane is a hyperplane that "just touches" the set C. If C is a convex set then there exists a supporting hyperplane for every point on its boundary. Supporting hyperplanes play a role in the definition of the subgradient of a non-differentiable function. A vector $\boldsymbol{\lambda}$ is called a *subgradient* of a convex proper function f at $\mathbf{x}$ if

$$\forall \mathbf{z} \quad f(\mathbf{z}) \geq f(\mathbf{x}) + \boldsymbol{\lambda}^\top(\mathbf{z} - \mathbf{x}). \tag{17}$$

This condition has a simple geometric meaning: it says that the affine function $h(\mathbf{z}) = f(\mathbf{x}) + \boldsymbol{\lambda}^\top(\mathbf{z} - \mathbf{x})$ is a (non-vertical) supporting hyperplane to the convex set epi(f) at the point $(\mathbf{x}, f(\mathbf{x}))$. Consequently, the set of subgradients $\boldsymbol{\lambda}$ at $\mathbf{x}$, called the *subdifferential of f at* $\mathbf{x}$ and is denoted by $\partial f(\mathbf{x})$, consists of the supporting hyperplanes to the convex set epi(f) at the point $(\mathbf{x}, f(\mathbf{x}))$. When f is differentiable at $\mathbf{x}$ then the supporting hyperplane is unique and $\partial f(\mathbf{x}) = \nabla f(\mathbf{x})$.

Definition 4: The sub-differential of a function f at a point $\mathbf{x}$ is denoted by $\partial f(\mathbf{x})$ and consists of all the supporting hyperplanes of epi(f) at the point $\mathbf{x}$, namely

$$\partial f(\mathbf{x}) = \{\boldsymbol{\lambda} : \forall \mathbf{z} \quad f(\mathbf{z}) \geq f(\mathbf{x}) + \boldsymbol{\lambda}^\top(\mathbf{z} - \mathbf{x})\}$$

The following claim describes the sub-differential of the indicator function associated with affine sets (a useful result which will serve us later):

Claim 3: Let A be $k \times n$ matrix and consider the affine set $\mathcal{B} = \{\mathbf{x} : A\mathbf{x} = \mathbf{c}\}$ and its indicator function

$$\delta_{\mathcal{B}}(\mathbf{x}) = \begin{cases} 0 & A\mathbf{x} = \mathbf{c} \\ \infty & otherwise \end{cases}$$

Then $\partial \delta_{\mathcal{B}} = \{A^\top \boldsymbol{\sigma} : \boldsymbol{\sigma} \in R^k\}$.

Proof: This claim results as a special case of [4] example 7.1.4. For the sake of clarity we provide a direct proof. We describe the sub-differential $\partial\delta_{\mathcal{B}}(\mathbf{x})$ for every point $\mathbf{x}$ in the domain of $\delta_{\mathcal{B}}$, i.e., $\delta_{\mathcal{B}}(\mathbf{x}) = 0$. To prove the direction $\partial\delta_{\mathcal{B}} \supseteq \{A^\top\boldsymbol{\sigma} : \boldsymbol{\sigma} \in R^k\}$ we must show that $\delta_{\mathcal{B}}(\mathbf{z}) \geq \delta_{\mathcal{B}}(\mathbf{x}) + \boldsymbol{\sigma}^\top A(\mathbf{z} - \mathbf{x})$ for every $\mathbf{z}$. For every $\mathbf{z}$ satisfying $A\mathbf{z} = \mathbf{c}$ this relation holds since $\delta_{\mathcal{B}}(\mathbf{z}) = 0$ and $A(\mathbf{z} - \mathbf{x}) = 0$. For every $\mathbf{z}$ with $A\mathbf{z} \neq \mathbf{c}$ this relation holds since $\delta_{\mathcal{B}}(\mathbf{z}) = \infty$.

To prove the other direction $\partial\delta_{\mathcal{B}} \subseteq \{A^\top\boldsymbol{\sigma} : \boldsymbol{\sigma} \in R^k\}$ we must show that $\delta_{\mathcal{B}}(\mathbf{z}) \geq \delta_{\mathcal{B}}(\mathbf{x}) + \boldsymbol{\lambda}^\top(\mathbf{z} - \mathbf{x})$ only if $\boldsymbol{\lambda} = A^\top\boldsymbol{\sigma}$ holds for every $\mathbf{z}$. First we note that the set $\{\mathbf{z} - \mathbf{x} : A\mathbf{z} = \mathbf{c}\}$ is orthogonal to $\{A^\top\boldsymbol{\sigma} : \boldsymbol{\sigma} \in R^k\}$, therefore if we assume on the contrary that $\boldsymbol{\lambda} \neq A^\top\boldsymbol{\sigma}$ there must be a vector $\mathbf{z}_0 - \mathbf{x}$ with non-vanishing angle with $\boldsymbol{\lambda}$, namely $\boldsymbol{\lambda}^\top(\mathbf{z}_0 - \mathbf{x}) > 0$ therefore Definition 4 does not hold for $\boldsymbol{\lambda}$. □

Claim 4: Consider a function f whose domain is contained in the affine set $\mathcal{B} = \{\mathbf{x} : A\mathbf{x} = \mathbf{c}\}$. Then whenever $\boldsymbol{\lambda} \in \partial f(\mathbf{x})$ there holds $(\boldsymbol{\lambda} + A^\top\boldsymbol{\sigma}) \in \partial f(\mathbf{x})$ for every $\boldsymbol{\sigma}$.

Proof: $f(\mathbf{x})$ can be equivalently written as $f(\mathbf{x}) + \delta_{\mathcal{B}}(\mathbf{x})$ where $\delta_{\mathcal{B}}$ is the indicator functions of the affine set $\mathcal{B}$, therefore $\partial f = \partial(f + \delta_{\mathcal{B}})$. From the linearity of the sub-differential, cf. [4] Theorem 4.2.4, there holds $\partial(f(\mathbf{x}) + \delta_{\mathcal{B}}(\mathbf{x})) = \partial f(\mathbf{x}) + \partial\delta_{\mathcal{B}}(\mathbf{x})$ and the claim follows since $\boldsymbol{\lambda} \in \partial f(\mathbf{x})$ by assumptions, and $A^\top\boldsymbol{\sigma} \in \partial\delta_{\mathcal{B}}(\mathbf{x})$ from Claim 3. □

A supporting hyperplane at $\mathbf{x}$ with $\boldsymbol{\lambda}$-slope must satisfy Definition 4, namely

$$\forall \mathbf{z} \quad f(\mathbf{z}) - \boldsymbol{\lambda}^\top\mathbf{z} \geq f(\mathbf{x}) - \boldsymbol{\lambda}^\top\mathbf{x},$$

therefore it must hold that the $\boldsymbol{\lambda}$-slope hyperplane supports the epigraph at $(\mathbf{x}, f(\mathbf{x}))$ where $\mathbf{x} \in \text{argmin}\{f(\mathbf{z}) - \boldsymbol{\lambda}^\top\mathbf{z}\}$. This leads to the definition of the *conjugate* function:

Definition 5: The Fenchel-Legendre conjugate is:

$$f^*(\boldsymbol{\lambda}) = \max_{\mathbf{x} \in dom(f)} \{(\boldsymbol{\lambda}^\top\mathbf{x} - f(\mathbf{x})\}.$$

The conjugate $f^*(\boldsymbol{\lambda})$ describes the offset of the $\boldsymbol{\lambda}$-hyperplane that supports the epigraph of f. Note that regardless of the structure of $f(\mathbf{x})$ its conjugate function $f^*(\boldsymbol{\lambda})$ is closed and convex, since it is the pointwise maximum of a collection of affine (closed) functions. Furthermore, if f is convex then the conjugate of its conjugate returns back f, i.e., $f^{**} = f$ (cf. [4], Theorem 7.1.1). The following claim is a useful result which shall serve us later:

Claim 5: Let $g^*(\boldsymbol{\lambda}) = f^*(\boldsymbol{\lambda} - \boldsymbol{\mu})$, then $g(\mathbf{x}) = f(\mathbf{x}) + \boldsymbol{\mu}^\top\mathbf{x}$

Proof: The definition of the Fenchel-Legendre conjugate of $g(\mathbf{x}) = f(\mathbf{x}) + \boldsymbol{\mu}^\top\mathbf{x}$ takes the form $g^*(\boldsymbol{\lambda}) = \max_x\{\boldsymbol{\lambda}^\top\mathbf{x} - \boldsymbol{\mu}^\top\mathbf{x} - f(\mathbf{x})\}$ which has the form in the claim since $\boldsymbol{\lambda}^\top\mathbf{x} - \boldsymbol{\mu}^\top\mathbf{x} = (\boldsymbol{\lambda} - \boldsymbol{\mu})^\top\mathbf{x}$ □

The convex conjugate plays an important role in duality. Consider contrained minimization under linear constraints $A\mathbf{x} = 0$, i.e., $\mathbf{a}_1^\top\mathbf{x} = 0, ..., \mathbf{a}_k^\top\mathbf{x} = 0$ with $\mathbf{a}_i^\top$ being the i'th row vector of A. The statement about the existence of Lagrange multiplier for non-differentiable functions is described next:

Theorem 3: **(Lagrange multipliers)**
Let $f(\mathbf{x})$ be a proper convex function and consider the convex program

$$\min_{\mathbf{x} \in dom(f)} f(\mathbf{x}) \quad \text{subject to} \quad A\mathbf{x} = 0.$$

Assume $ri(dom(f))$ intersect the linear constraints $A\mathbf{x} = 0$ and that the optimal value of the program is finite. Then there exists Lagrange multipliers $\lambda_1^*, ..., \lambda_k^*$ satisftying

$$\mathbf{x}^* \in \underset{\mathbf{x} \in dom(f)}{\text{argmin}} \{f(\mathbf{x}) + \sum_i \lambda_i^*\mathbf{a}_i\} \tag{18}$$

Proof: The assumptions 6.4.1 in [4] hold in this case and following the Nonlinear Farkas lemma, as done in Theorem 6.4.2 in [4], completes the proof. □

The duality theorem using the conjugate f^* is described below:

Theorem 4: **(Strong Duality)** Let f be a convex proper function and $ri(dom(f))$ intersects with the constraints $A\mathbf{x} = 0$, and that the optimal value of the program is finite. The following form a primal-dual pair:

$$(primal) \qquad \min_{\mathbf{x} \in domain(f)} f(\mathbf{x}) \quad \text{s.t.} \quad A\mathbf{x} = 0 \tag{19}$$

$$(dual) \qquad \max_{\boldsymbol{\lambda} = \lambda_1, ..., \lambda_k} -f^*(-A^\top\boldsymbol{\lambda}) \tag{20}$$

Then there is no duality gap and there exists primal-dual optimal pair. Moreover, the vectors $(\mathbf{x}^*, \boldsymbol{\lambda}^*)$ form a primal-dual optimal pair $f(\mathbf{x}^*) = -f^*(-A^\top\boldsymbol{\lambda}^*)$ if and only if the following "algorithmic certificate" for optimality hold:

$$\begin{aligned} &\mathbf{x}^* \in dom(f) && \text{(feasibility)} \\ &\mathbf{0} \in \partial\{f(\mathbf{x}^*) + A^\top\boldsymbol{\lambda}^*\} && \text{(optimality)} \end{aligned}$$

Proof: The existence of primal-dual optimal pair follows from Theorem 3. The rest follows from [4], Theorem 6.2.5 □

Note that due to the linearity of the sub-differential $\partial(f+g) = \partial f + \partial g$, the optimality condition above is equivalent to $-A^\top \boldsymbol{\lambda}^* \in \partial f(\mathbf{x}^*)$.

To see the connection to Lagrangian duality, note that by definition of f^* we have:

$$-f^*(-A^\top \boldsymbol{\lambda}^*) = \min_{\mathbf{x}\in dom(f)} \{f(\mathbf{x}) + A^\top \boldsymbol{\lambda}^*\},$$

which in turn means that the primal-dual pair $(\mathbf{x}^*, \boldsymbol{\lambda}^*)$ satisfy $\mathbf{x}^* \in \text{argmin}_\mathbf{x}\{f(\mathbf{x}) + A^\top \boldsymbol{\lambda}^*\}$ where the right-hand side is the Lagrangian $L(\mathbf{x}, \boldsymbol{\lambda}) = f(\mathbf{x}) + A^\top \boldsymbol{\lambda}$ and the dual problem is $\max_{\boldsymbol{\lambda}} q(\boldsymbol{\lambda})$ where $q(\boldsymbol{\lambda}) = \min_\mathbf{x} L(\mathbf{x}, \boldsymbol{\lambda})$.

A proper convex function $f(\mathbf{x})$ is *essentially strictly convex* if it is strictly convex on every convex subset in $dom(\partial f)$. We note below that in order for the dual function to be smooth the primal must be strictly convex. A smooth dual is necessary for a dual ascent scheme (described later).

Theorem 5: **(strict primal $\Longleftrightarrow$ smooth dual)**
A closed proper convex function is essentially strictly convex if and only if its conjugate it essentially smooth.

Proof: [46], Theorem 26.3 □

We describe below two Fenchel duality theorems which are the functional form of the Lagrange duality where the constraints are implicit in the functions domains:

Theorem 6: **Basic Fenchel Duality I**
Let $g(\mathbf{x}), h(\mathbf{x})$ be proper closed and convex functions and $ri(dom(g)) \cap ri(dom(h)) \neq \emptyset$, and the value of the program is finite. The following are primal and dual programs:

$$\begin{aligned} \text{Primal:} \quad & \min_\mathbf{x} g(\mathbf{x}) + h(\mathbf{x}) \\ \text{Dual:} \quad & \max_{\boldsymbol{\lambda}} -g^*(-\boldsymbol{\lambda}) - h^*(\boldsymbol{\lambda}) \end{aligned}$$

Then there is no duality gap, and there exists primal-dual optimal pair. Moreover, the vectors $(\mathbf{x}^*, \boldsymbol{\lambda}^*)$ are primal-dual optimal pair if and only if $-\boldsymbol{\lambda}^* \in \partial g(\mathbf{x}^*)$ and $\boldsymbol{\lambda}^* \in \partial h(\mathbf{x}^*)$. Conversely, by reversing the roles of primal and dual, the vectors $(\mathbf{x}^*, \boldsymbol{\lambda}^*)$ are primal-dual optimal pair if and only if $\mathbf{x}^* \in \partial g^*(-\boldsymbol{\lambda}^*)$ and $\mathbf{x}^* \in \partial h^*(\boldsymbol{\lambda}^*)$. In particular, if $g(\mathbf{x})$ is essentially strictly convex and $g^*(\boldsymbol{\lambda})$ is finite, then the optimal $\mathbf{x}^*$ is determined by $\mathbf{x}^* \in \nabla g^*(-\boldsymbol{\lambda}^*)$.

Proof: We reduce Fenchel duality to Lagrange duality in Theorem 4, where we consider a decomposed version of the primal function $f(\mathbf{x}_g, \mathbf{x}_h) = g(\mathbf{x}_g) + h(\mathbf{x}_h)$ subject to the linear consistency constraints $\mathbf{x}_g = \mathbf{x}_h$. Note that the vector equality constraint is composed from m equality constraints where m is the length of the vectors $\mathbf{x}_g$ and $\mathbf{x}_h$, therefore we expect to use Lagrange multipliers vector $\boldsymbol{\lambda}$ of length n. The Lagrangian $L(\mathbf{x}_g, \mathbf{x}_h, \boldsymbol{\lambda})$ takes the form $g(\mathbf{x}_g) + h(\mathbf{x}_h) + \boldsymbol{\lambda}^\top(\mathbf{x}_g - \mathbf{x}_h)$ and using the conjugate notation in Definition. 5 the dual function $q(\boldsymbol{\lambda}) = \min_{\mathbf{x}_g, \mathbf{x}_h} L()$ takes the form in the theorem above. Following Theorem 4 there exists primal-dual optimal pair which must satisfy the feasibility condition, i.e. $\mathbf{x}_g^* = \mathbf{x}_h^*$, and the optimality condition, namely $-\boldsymbol{\lambda}^* \in \partial g(\mathbf{x}_g^*)$ and $\boldsymbol{\lambda}^* \in \partial h(\mathbf{x}_h^*)$. The theorem follows as the optimal $\mathbf{x}^*$ must equal $\mathbf{x}_g^*$ as well as $\mathbf{x}_h^*$. Reversing the roles of primal and dual are allowed by convexity whereby $g^{**} = g, h^{**} = h$. Furthermore, since $g^*(\boldsymbol{\lambda})$ is finite Theorem 5 determines $g^*(\boldsymbol{\lambda})$ to be smooth, and whenever $\mathbf{x}^* \in \partial g^*(-\boldsymbol{\lambda}^*)$ there must hold $\mathbf{x}^* \in \nabla g^*(-\boldsymbol{\lambda}^*)$. □

The next theorem is generalizes the Fenchel duality theorem above:

Theorem 7: **Basic Fenchel Duality II**
Let $f(\mathbf{x}), h_1(\mathbf{x}), ..., h_n(\mathbf{x})$ be proper, closed and convex functions and $ri(dom(f)) \cap ri(dom(h_i)) \neq \emptyset$ and the optimal value of the program is finite. The following are primal and dual programs:

$$\begin{aligned} \text{Primal:} \quad & \min_\mathbf{x} f(\mathbf{x}) + \sum_{i=1}^n h_i(\mathbf{x}) \\ \text{Dual:} \quad & \max_{\boldsymbol{\lambda}} \{-f^*(-\sum_{i=1}^n \boldsymbol{\lambda}_i) - \sum_{i=1}^n h_i^*(\boldsymbol{\lambda})\} \end{aligned} \tag{21}$$

Then there is no duality gap, and there exists prima-dual optimal pair. Moreover, the vectors $(\mathbf{x}^*, \boldsymbol{\lambda}_i^*)$ are primal-dual optimal pair if and only if $-\sum_{i=1}^n \boldsymbol{\lambda}_i^* \in \partial f(\mathbf{x}^*)$ and $\boldsymbol{\lambda}_i^* \in \partial h_i(\mathbf{x}^*)$. Also, if $f(\mathbf{x})$ is essentially strictly convex and $f^*(\boldsymbol{\lambda})$ is finite, then $\mathbf{x}^* = \nabla f^*(-\sum_i \boldsymbol{\lambda}_i^*)$.

Proof: The proof closely follows the one of Theorem 6 where we consider a decomposed version of the primal function $f(\mathbf{x}_f) + \sum_i h_i(\mathbf{x}_i)$ subject to the linear consistency constraints $\mathbf{x}_f = \mathbf{x}_i$. The Lagrangian $L()$ takes the form $f(\mathbf{x}_f) + \sum_i h_i(\mathbf{x}_i) + \sum_i \boldsymbol{\lambda}_i^\top(\mathbf{x}_f - \mathbf{x}_i)$ and the dual function $q(\boldsymbol{\lambda}_i)$ takes the form in the theorem above. Following the Lagrange duality in Theorem 4 there exists primal-dual optimal pair which must be primal feasible, i.e., $\mathbf{x}_f^* = \mathbf{x}_i^*$, and satisfy the optimality condition $-\sum_{i=1}^n \boldsymbol{\lambda}^* \in \partial f(\mathbf{x}_f^*)$ and $\boldsymbol{\lambda}_i^* \in \partial h_i(\mathbf{x}_i^*)$. Whenever $f(\mathbf{x})$ is essentially strictly convex and $f^*(\boldsymbol{\lambda})$ is finite, repeating the primal-dual reversing argument of Theorem 6 shows that $\mathbf{x}^* = \nabla f^*(-\sum_i \boldsymbol{\lambda}_i^*)$. □

Algorithmically, minimizing the primal program $f(\mathbf{x}) + \sum_{i=1}^n h_i(\mathbf{x})$ requires to take into account the domains of f and h_i simultaneously. Therefore, it is algorithmically appealing to solve the primal program in a piece-meal fashion using dual block ascent, while iteratively improving a single vector $\boldsymbol{\lambda}_i$. This way one need to consider only sub-problems that consists of f^* and a single h_i^*. After we recover the optimal $\boldsymbol{\lambda}_i^*$ one can recover efficiently the primal optimal $\mathbf{x}^*$ by using the smoothness of f^* as describes in Theorem 7:

Algorithm 6 (Dual Block Coordinate Ascent): Initialize $\boldsymbol{\lambda}_1 = \mathbf{0}, ..., \boldsymbol{\lambda}_n = \mathbf{0}$.

```
1) Repeat until convergence:
2) For $i = 1, ...n$:
    a) $\boldsymbol{\mu}_i \leftarrow \sum_{j\neq i} \boldsymbol{\lambda}_j$
    b) $\boldsymbol{\lambda}_i \leftarrow \underset{\boldsymbol{\lambda}_i}{\text{argmax}} \{-f^*(-\boldsymbol{\lambda}_i - \boldsymbol{\mu}_i) - h_i^*(\boldsymbol{\lambda}_i)\}$
Output $\mathbf{x}^* = \nabla f^*(-\sum_i \boldsymbol{\lambda}_i^*)$.
```

The dual block ascent algorithm iteratively improves the dual objective therefore is guaranteed to converge. Whenever $f(\mathbf{x})$ is strictly convex in its domain its conjugate is essentially smooth and the dual block ascent is guaranteed to converge to the global optimum, as formally described below:

Theorem 8: **(Dual Block Ascent)** Let f, h_i be closed convex functions and assume the relative interior of their domains intersect. In addition, assume h_i are continuous over their domains and f is strictly convex over its domain and f^* is finite. Then, the dual block ascent algorithm converges to the dual and primal optimum.

In particular, if the dual sequence is bounded then every of its limit points is an optimal dual solution $\boldsymbol{\lambda}_1^*, ..., \boldsymbol{\lambda}_n^*$. Also, consider the primal sequence generated by $\nabla f^*(-\sum_i \boldsymbol{\lambda}_i)$ computed from the dual sequence, then this primal sequence is bounded and its limit point is the optimal solution $\mathbf{x}^*$.

Proof: [36]. □

Appendix B
The Primal-Dual Block Ascent Algorithm

We describe an algorithm for solving programs of the form

$$f(\mathbf{b}) + \sum_i h_i(\mathbf{b})$$

while solving sub-problems which consists of $f(\mathbf{b})$ and a single function $h_i(\mathbf{b})$. In our framework we include convex as well as non-convex optimization, but for now we describe the convex settings, and later describe the necessary conditions for this optimization scheme for non-convex programs. The dual block ascent method, described in Algorithm 6 decomposes the optimization program to sub-problems which solve a dual function which requires the explicit computation of the conjugate functions $f^*(\boldsymbol{\lambda})$ and $h_i^*(\boldsymbol{\lambda})$ — a task which is often algorithmically unattractive or unfeasible. Instead, one can recover $\boldsymbol{\lambda}_i$ in Algorithm 6 by solving its primal program and using the primal-dual optimality condition in Theorem 6 as follows. Set $h(\mathbf{b}) \leftarrow h_i(\mathbf{b})$ and $g(\mathbf{b}) \leftarrow f(\mathbf{b}) + \mathbf{b}^\top \boldsymbol{\mu}_i$ and recall Claim 5 from which we obtain $g^*(-\boldsymbol{\lambda}_i) = f^*(-\boldsymbol{\lambda}_i - \boldsymbol{\mu}_i)$, and solve the primal program:

$$\mathbf{b}^* = \underset{\mathbf{b} \in dom(f) \cap dom(h_i)}{\operatorname{argmin}} \left\{ f(\mathbf{b}) + \mathbf{b}^\top \boldsymbol{\mu}_i + h_i(\mathbf{b}) \right\} \quad (22)$$

If the pair of functions $f(\mathbf{b})$ and $h_i(\mathbf{b})$ satisfy the assumptions of Theorem 6 then the functions $g(\mathbf{b}) \leftarrow f(\mathbf{b}) + \mathbf{b}^\top \boldsymbol{\mu}_i$ and $h_i(\mathbf{b})$ satisfy these assumptions as well and, hence, $\boldsymbol{\lambda}_i$ can be recovered from the optimality conditions of Theorem 6:

$$\boldsymbol{\lambda}_i^* \in \{-\boldsymbol{\mu}_i - \partial f(\mathbf{b}^*)\} \cap \partial h_i(\mathbf{b}^*) \quad (23)$$

Taken together, one obtains the primal form of the dual block ascent algorithm, in which one need not compute the conjugate functions:

Algorithm 7 (Primal-Dual Vanilla): Let the functions $f(\mathbf{b})$ and $h_i(\mathbf{b})$ satisfy the conditions of Theorem 8. Initialize $\boldsymbol{\lambda}_1 = \mathbf{0}, ..., \boldsymbol{\lambda}_n = \mathbf{0}$.

```
1) Repeat until convergence:
2) For $i = 1, ...n$:
    a) $\boldsymbol{\mu}_i \leftarrow \sum_{j \neq i} \boldsymbol{\lambda}_j$
    b) $\mathbf{b}^* \leftarrow \underset{\mathbf{b} \in dom(f) \cap dom(h_i)}{\operatorname{argmin}} \{ f(\mathbf{b}) + h_i(\mathbf{b}) + \mathbf{b}^\top \boldsymbol{\mu}_i \}$.
    c) Recover $\boldsymbol{\lambda}_i \in \{-\boldsymbol{\mu}_i - \partial f(\mathbf{b}^*)\} \cap \partial h_i(\mathbf{b}^*)$
Output $\mathbf{b}^*$.
```

A useful property of the primal-dual algorithm (and its special cases described in the sequel) is that the sparseness structure of $\boldsymbol{\lambda}_i$ conforms to the local structure of the functions h_i in the following sense: assume the variables $\mathbf{b}$ are indexed by $1, ..., m$, and the function $h_i(\mathbf{b})$ depends on small subset of variables indexed by $N(i) \subset \{1, ..., m\}$, then $\boldsymbol{\lambda}_{i,\alpha}^*$ contains information only for $\alpha \in N(i)$ and the remaining entries vanish.

Claim 6 (Locality of Dual Variables): Assume variables $\mathbf{b}$ are indexed by $\{1, ..., m\}$ and $h_i(\mathbf{b})$ depends only on a subset of variables indexed by $N(i) \subset \{1, ..., m\}$, then the following hold:

$$\boldsymbol{\lambda} \in \partial h_i(\mathbf{b}) \qquad \Longrightarrow \qquad \forall \beta \notin N(i) \quad \boldsymbol{\lambda}_\beta = 0$$

Proof: Consider the decomposition of $\mathbf{b}$ to two parts $\mathbf{b} = (\mathbf{b}_{N(i)}, \mathbf{b}_{\bar{N}(i)})$, where $\bar{N}(i)$ is the complement $\{1, ..., m\} \setminus N(i)$, and likewise for the sub-gradient $\boldsymbol{\lambda}$. Following Definition 4 if $\boldsymbol{\lambda} \in \partial h(\mathbf{b})$ then

$$h_i(\hat{\mathbf{b}}) \geq h_i(\mathbf{b}) + \boldsymbol{\lambda}^\top (\hat{\mathbf{b}} - \mathbf{b}) \quad \forall \hat{\mathbf{b}}. \quad (24)$$

The linear term $\boldsymbol{\lambda}^\top(\hat{\mathbf{b}} - \mathbf{b})$ decomposes to the sum of $\boldsymbol{\lambda}_{N(i)}^\top(\hat{\mathbf{b}}_{N(i)} - \mathbf{b}_{N(i)})$ and $\boldsymbol{\lambda}_{\bar{N}(i)}^\top(\hat{\mathbf{b}}_{\bar{N}(i)} - \mathbf{b}_{\bar{N}(i)})$. Since $\hat{\mathbf{b}}$ is arbitrary we can choose $\hat{\mathbf{b}} = (\hat{\mathbf{b}}_{N(i)}, \hat{\mathbf{b}}_{\bar{N}(i)})$ where $\hat{\mathbf{b}}_{N(i)}$ is set to $\hat{\mathbf{b}}_{\bar{N}(i)} = r(\boldsymbol{\lambda}_{\bar{N}(i)} - \mathbf{b}_{\bar{N}(i)})$ for some arbitrary scalar $r > 0$, and $\hat{\mathbf{b}}_{\bar{N}(i)}$ is arbitrary. Eqn. 24 then becomes:

$$h_i(\hat{\mathbf{b}}) \geq h_i(\mathbf{b}) + \boldsymbol{\lambda}_{N(i)}^\top(\hat{\mathbf{b}}_{N(i)} - \mathbf{b}_{N(i)}) + r\boldsymbol{\lambda}_{\bar{N}(i)}^\top \boldsymbol{\lambda}_{\bar{N}(i)},$$

for all $r > 0$. If we assume to the contrary that $\boldsymbol{\lambda}_{\bar{N}(i)} \neq \mathbf{0}$ then we can increase the value of r and thus make the righ-hand side of the equation arbitrarily high, while not effecting the left hand side since $h_i(\hat{\mathbf{b}})$ is independent of r by the claim assumption (as h_i depends only on the variables indexed by $N(i)$) - in contradiction to $\boldsymbol{\lambda} \in \partial h(\mathbf{b})$. □

The primal-dual algorithm is still unattractive as it requires the evaluation of the sub-differentials of ∂f and ∂h_i which could be as difficult as the computation of the conjugate functions. Our setting, however, is more constrained than the setting described in Theorem 8. In particular, the function $f = \hat{f} + \delta_{\mathcal{B}}$ where $\hat{f}$ is essentially smooth and $\mathcal{B} = \{\mathbf{b} \,:\, A\mathbf{b} = \mathbf{c}\}$ is an affine set. Since f is non-differentiable the dual is not strictly convex, and thus we cannot expect $\boldsymbol{\lambda}_i$ to be uniquely defined. Nevertheless, we show below that $\boldsymbol{\lambda}_i$ has a convenient and simple form.

Claim 7: Let $f(\mathbf{b}) = \hat{f}(\mathbf{b}) + \delta_{\mathcal{B}}(\mathbf{b})$ where $\hat{f}$ is essentially smooth and $\mathcal{B} = \{\mathbf{b} \,:\, A\mathbf{b} = \mathbf{c}\}$ and assume that $dom(h_i) \subseteq dom(f)$. Assume the functions $g(\mathbf{b}) \leftarrow f(\mathbf{b}) + \mathbf{b}^\top \boldsymbol{\mu}_i$ and $h(\mathbf{b}) \leftarrow h_i(\mathbf{b})$ satisfy the assumptions of Theorem 6. Then for every real vector $\boldsymbol{\sigma}$ the sub-gradient $\boldsymbol{\lambda}_i^* = -\boldsymbol{\mu}_i - \nabla f_s(\mathbf{b}^*) + A^\top \boldsymbol{\sigma}$ is optimal dual, i.e. satisfies Eqn. 23

Proof: Theorem 6 ensures the existence of a primal-dual pair $(\mathbf{b}^*, \boldsymbol{\lambda}^*)$ which satisfy Eqn. 23. The domains of $f(\mathbf{b})$ and $h_i(\mathbf{b})$ are contained in $\mathcal{B}$ by assumption, therefore by Claim 4

$$\forall \boldsymbol{\sigma} \qquad \boldsymbol{\lambda}_i^* + A^\top \boldsymbol{\sigma} \in \{-\boldsymbol{\mu}_i - \partial f(\mathbf{b}^*)\} \cap \partial h_i(\mathbf{b}^*),$$

meaning that for every $\boldsymbol{\sigma}$ the sub-gradient $(\boldsymbol{\lambda}_i^* + A^\top \boldsymbol{\sigma})$ is dual optimal. From linearity of the sub-differential we have $\partial f(\mathbf{b}^*) = \nabla \hat{f}(\mathbf{b}^*) + \partial \delta_{\mathcal{B}}$. Following Claim 3 the sub-differential $\delta_{\mathcal{B}}$ is represented by vectors in the linear subspace spanned by the columns of $A^\top$, denoted by $A^\top \boldsymbol{\sigma}_0$. Using again the linearity of the sub-differential we deduce

$$\forall \boldsymbol{\sigma} \qquad (\boldsymbol{\lambda}_i^* + A^\top \boldsymbol{\sigma}) = -\boldsymbol{\mu}_i - \nabla \hat{f}(\mathbf{b}^*) + A^\top(\boldsymbol{\sigma} - \boldsymbol{\sigma}_0)$$

is a dual optimal sub-gradient. The claim follows by replacing $\boldsymbol{\lambda}_i^* \leftarrow (\boldsymbol{\lambda}_i^* + A^\top \boldsymbol{\sigma})$. □

Algorithm 8 (Primal-Dual Ascent): Let the functions $f(\mathbf{b})$ and $h_i(\mathbf{b})$ satisfy the conditions of Theorem 8 where in addition let $f(\mathbf{b}) = \hat{f}(\mathbf{b}) + \delta_{\mathcal{B}}(\mathbf{b})$ where $\hat{f}(\mathbf{b})$ is essentially smooth, $\mathcal{B} = \{\mathbf{b} : A\mathbf{b} = \mathbf{c}\}$ and $dom(h_i) \subseteq dom(f)$. Initialize $\boldsymbol{\lambda}_1 = \mathbf{0}, ..., \boldsymbol{\lambda}_n = \mathbf{0}$.

```
1) Repeat until convergence:
2) For $i = 1, ...n$:
    a) $\boldsymbol{\mu}_i \leftarrow \sum_{j \neq i} \boldsymbol{\lambda}_j$
    b) $\mathbf{b}^* \leftarrow \operatorname{argmin}_{\mathbf{b} \in dom(f) \cap dom(h_i)} \left\{ f(\mathbf{b}) + h_i(\mathbf{b}) + \mathbf{b}^\top \boldsymbol{\mu}_i \right\}$
    c) $\boldsymbol{\lambda}_i \leftarrow -\boldsymbol{\mu}_i - \nabla \hat{f}(\mathbf{b}^*) + A^\top \boldsymbol{\sigma}$
Output $\mathbf{b}^*$.
```

Claim 8 (Convergence): Algorithm 8 converges to the dual and primal optimum. Moreover, its primal sequence converges to the primal optimal point $\mathbf{b}^*$ and whenever its dual sequence is bounded every of its limit point is an optimal dual solution $\boldsymbol{\lambda}_1^*, ..., \boldsymbol{\lambda}_n^*$.

Proof: Algorithm 8 implicitly performs dual block ascent and the dual sequence it generates is identical to the dual sequence generated by Algorithm 6 therefore inherits the features described in Theorem 8. Theorem 6 relates $\mathbf{b}^*$ with the primal sequence describe in Theorem 8 by $\mathbf{b}^* = \nabla f^*(-\boldsymbol{\lambda}_i - \boldsymbol{\mu}_i)$ □

The special case of Algorithm 8 when $h_i = \delta_{C_i}$, where C_i is a convex set, and f is essentially smooth, i.e., $A = 0$, can be mapped (by eliminating step 2(a)) to a *successive Bregman projection* algorithm [6], [7] which is also known under the names of Dykstra, Hildreth, Han and Csiszar. This class of iterative projection schemes has a long history starting from Von-Neumann in the 50s [43] who introduced the case where $f(\mathbf{b}) = \|\mathbf{b} - \mathbf{b}_0\|^2$ and C_i are affine sets. In that case the primal solution is to find the projection of $\mathbf{b}_0$ onto the intersection of the affine sets $C_1 \cap ... \cap C_n$ and the sub-problem in Eqn. 22 corresponds to the projection of $\boldsymbol{\mu}_i$ onto the affine set C_i. Hildreth [23] extended the problem with open half spaces $C_i = \{\mathbf{x} \mid \mathbf{a}_i^\top \mathbf{x} \leq b_i\}$. Bregman [5] extended Hildreth's problem setup by including any strictly convex function f. The special case of Entropy projections was introduced later by Csiszar [10], as I-projections. Dykstra [12], [11] was the first to introduce general convex sets C_i (i.e., going beyond affine sets or half-spaces) but limited the treatment to f representing the Euclidean norm and the KL divergence. The view of the algorithm with general essentially smooth f and convex sets C_i as performing successive Bregman projections is due to [7], [6].

Algorithm 8 extends the body of iterative schemes mentioned above along three directions: (i) f is extended to non-smooth functions which in turn makes $\boldsymbol{\lambda}_i$ non-uniquely defined, (ii) as a result $\boldsymbol{\lambda}_i$ is defined up to an additive term which in the context of the message-passing norm-product algorithm (and its special cases) translates to the normalization of the messages $n_{i \to \alpha}$, and (iii) our algorithm has two auxiliary variables, $\boldsymbol{\mu}_i$ and $\boldsymbol{\lambda}_i$, which allows a straightforward mapping onto a message-mapping framework and complies with the local structure of the underlying graph (Claim 1).

A. The non-convex case

So far both f and h_i were convex, yet Algorithm 8 is still well defined when the functions h_i are non-convex. The purpose of this section is to clarify what can be guaranteed under such conditions. We will show that indeed there is no convergence guarantees, but if the algorithm does converge then it will do so to a stationary point of the primal program.

To minimize the program $f(\mathbf{b}) + \sum_i h_i(\mathbf{b})$ one must introduce Lagrange multipliers $\boldsymbol{\lambda}_1, ..., \boldsymbol{\lambda}_n$. Whenever $f(\mathbf{b})$ and $h_i(\mathbf{b})$ are convex the Lagrange multipliers are the arguments of the dual function, and recovering $\boldsymbol{\lambda}_i^*$ amounts to improving the dual objective with its best $\boldsymbol{\lambda}_i$-arguments, therefore this procedure is guaranteed to converge. When h_i are non-convex the Lagrange multipliers *do not* correspond to a dual function, and thus recovering $\boldsymbol{\lambda}_i^*$ amounts to finding a stationary point with respect to a sub-problem involving $f(\mathbf{b})$ and a single $h_i(\mathbf{b})$, and convergence cannot be guaranteed in general. Nevertheless, in each iteration we recover Lagrange multipliers for a stationary point of a related sub-problem, therefore, intuitively, *if* this method converges, it reaches a stationary point of the non-convex program $f(\mathbf{b}) + \sum_i h_i(\mathbf{b})$.

We consider programs with non-convex smooth functions $h_i(\mathbf{b})$ restricted to the affine domain $\{\mathbf{b} : A_i \mathbf{b} = \mathbf{c}_i\}$, and Legendre-type function $f(\mathbf{b})$, whose conjugate function is finite. Recall Theorem 2, describing Legendre-type function as an essentially smooth function which is strictly convex in its interior and satisfies $\nabla f^* = (\nabla f)^{-1}$. For this type of non-convex programs we show in the following claim, that if Algorithm 8 converges, it reaches a local-minimum of $f(\mathbf{b}) + \sum_i h_i(\mathbf{b})$:

Claim 9: Consider Algorithm 8 with $A = 0$ for Legendre-type function $f(\mathbf{b})$ and non-convex continuously differentiable functions $h_i(\mathbf{b})$ restricted to the affine domain $\{\mathbf{b} : A_i \mathbf{b} = \mathbf{c}_i\}$, and assume $\mathbf{b}^*$ in Eqn. 22 is in the interior of $dom(f)$ relative to the affine set $dom(h_i)$. Then if the algorithm converges it reaches a stationary point of the non-convex program $f(\mathbf{b}) + \sum_i h_i(\mathbf{b})$.

Proof: The optimization

$$\mathbf{b}^{*,(i)} = \operatorname*{argmin}_{\mathbf{b} \in dom(h_i)} \left\{ f(\mathbf{b}) + \mathbf{b}^\top \boldsymbol{\mu}_i + h_i(\mathbf{b}) \right\}$$

satisfies the conditions of the Lagrange multiplier Theorem 4 with respect to the affine set $dom(h_i)$, therefore if algorithm converges there holds

$$(*)\ \forall i \qquad \nabla f(\mathbf{b}^{*,(i)}) + \boldsymbol{\mu}_i + \nabla h_i(\mathbf{b}^{*,(i)}) + A_i^\top \boldsymbol{\nu}_i^* = 0$$

From steps 2a and 2c, for every i there must hold $\sum_{j=1}^n \boldsymbol{\lambda}_j = -\nabla f(\mathbf{b}^{*,(i)})$. The conjugate of Legendre-type

$$\min_{b_i,b_\alpha,\alpha\in N(i)} \left\{ -\sum_{x_i} b_i(x_i)\ln\phi_i(x_i) - \sum_{\alpha\in N(i)}\sum_{\mathbf{x}_\alpha} b_\alpha(\mathbf{x}_\alpha)\ln\hat{\psi}_{i,\alpha}(\mathbf{x}_\alpha) - \epsilon\hat{c}_i H(\mathbf{b}_i) - \sum_{\alpha\in N(i)}\epsilon\hat{c}_{i\alpha}(H(\mathbf{b}_\alpha)-H(\mathbf{b}_i)) \right\} \quad (25)$$

$$subject\ to: \quad \sum_{\mathbf{x}_\alpha} b_\alpha(\mathbf{x}_\alpha)=1,\ \sum_{\mathbf{x}_\alpha\setminus x_i} b_\alpha(\mathbf{x}_\alpha)=b_i(x_i),\quad \forall\mathbf{x}_\alpha,\alpha\in N(i)$$

Fig. 7. The local sub-problem of in step (b) of Algorithm 1 ($\min f_\epsilon(\mathbf{b})+\mathbf{b}^\top\boldsymbol{\mu}_i+h_{\epsilon,i}(\mathbf{b})$) solved by the norm-product algorithm.

function satisfies $\nabla f^* = (\nabla f)^{-1}$ by Theorem 2, therefore for every i holds $\mathbf{b}^{*,(i)} = \nabla f^*(-\sum_j \boldsymbol{\lambda}_i)$. This implies that the local primal arguments $\mathbf{b}^{*,(i)}$ are the same for every i, and we denote them by $\mathbf{b}^*$. Summing up the relations in (*) we get

$$\sum_i \nabla h_i(\mathbf{b}^*) + n\nabla f(\mathbf{b}^*) + \sum_{i=1}^n \boldsymbol{\mu}_i + \sum_{i=1}^n A_i\boldsymbol{\nu}_i^* = 0.$$

Substituting $\boldsymbol{\mu}_i = -\boldsymbol{\lambda}_i - \nabla f(\mathbf{b}^*)$ (from step 2c) we obtain the stationary condition for $\mathbf{b}^*$, i.e. $\nabla f(\mathbf{b}^*) + \sum_i \nabla h_i(\mathbf{b}^*) + \sum_{i=1}^n A_i\boldsymbol{\nu}_i^* = 0$. $\square$

B. The non-strictly convex case

The case $\epsilon = 0$ in eqn. 7 corresponds to having a non-strictly convex function f_ϵ in eqn. 10. This situation can be analyzed in greater generality by observing the behavior of Algorithm 7 when the function f is convex but not strictly convex.

For convex $f(\mathbf{b})$ and $h_i(\mathbf{b})$ the primal program in Eqn. 19 upper bounds the dual function in Eqn. 20, and the dual block ascent optimization scheme which iteratively improves the dual function must converge. If the function $f(\mathbf{b})$ is not strictly convex its conjugate is not smooth and the dual block ascent is not guaranteed to reach the global optimum. We describe, in a nutshell, where things go wrong in the Algorithm 7: Assume the algorithm converges, then for every i we obtain the primal solution

$$\mathbf{b}^{*,(i)} = \operatorname*{argmin}_{\mathbf{b}\in dom(f)\cap dom(h_i)} \left\{ f(\mathbf{b}) + \mathbf{b}^\top\boldsymbol{\mu}_i^* + h_i(\mathbf{b}) \right\}$$

Recovering the dual variables corresponds to finding $\boldsymbol{\lambda}_i^* \in \{-\boldsymbol{\mu}_i^* - \partial f(\mathbf{b}^{*,(i)})\}$. Recall that $\boldsymbol{\mu}_i^* = \sum_{j\neq i}\boldsymbol{\lambda}_j^*$, then the primal-dual relation boils down to $-\sum_j \boldsymbol{\lambda}_j^* = \partial f(\mathbf{b}^{*,(i)})$ for every i. If f was strictly convex it would have imply that all the $\mathbf{b}^{*,(i)}$ are in fact the same $\mathbf{b}^*$ and it would have ensure optimality. Since $f(\mathbf{b})$ is not strictly convex it means that the algorithm might converge in the dual domain but we cannot recover a consistent $\mathbf{b}^*$.

APPENDIX C
THE NORM-PRODUCT ALGORITHM

We embed the function definitions of f_ϵ and $h_{\epsilon,i}$ into the primal-dual Algorithm 1. Given the sparse structure of $h_{\epsilon,i}$ then, following Claim 1, we present the entries of $\boldsymbol{\lambda}_i$ according to the factor-graph structure by setting $\boldsymbol{\lambda}_i = \{\lambda_{i,\alpha}(\mathbf{x}_\alpha)\}$ (and likewise $\boldsymbol{\mu}_{i,\alpha}$). We first define few short-cut notations:

$$\begin{aligned} \hat{\psi}_{i,\alpha}(\mathbf{x}_\alpha) &\overset{def}{=} \psi_\alpha(\mathbf{x}_\alpha)\exp(-\mu_{i,\alpha}(\mathbf{x}_\alpha)), \quad (27)\\ \hat{c}_{i\alpha} &\overset{def}{=} c_\alpha + c_{i\alpha},\\ \hat{c}_i &\overset{def}{=} c_i + \sum_{\alpha\in N(i)} c_\alpha \end{aligned}$$

Step (b) of Algorithm 1 is reduced to finding $\mathbf{b}_\alpha^*$ for all $\alpha \in N(i)$, described in eqn. 25 in Fig. 7:

We will derive the optimal $\mathbf{b}_\alpha^*$ and show it has a closed-form solution. In the process we will be relying on the following observation which we present as a Lemma, without a proof:

Lemma 1: Let ψ be a non-negative array and $\mathbf{p}^*$ be the optimal probability array for the following optimization problem:

$$\mathbf{p}^* = \operatorname*{argmin}_{p(\mathbf{x})\geq 0,\sum_{\mathbf{x}} p(\mathbf{x})=1} \left\{ -\sum_{\mathbf{x}} p(\mathbf{x})\ln\psi(\mathbf{x}) - H(\mathbf{p}) \right\},$$

then,

$$\begin{aligned} p^*(\mathbf{x}) &= \frac{1}{\sum_{\mathbf{y}}\psi(\mathbf{y})}\psi(\mathbf{x}) \quad (28)\\ -\ln\sum_{\mathbf{x}}\psi(\mathbf{x}) &= -\sum_{\mathbf{x}} p^*(\mathbf{x})\ln\psi(\mathbf{x}) - H(\mathbf{p}^*) \quad (29) \end{aligned}$$

$\square$

We will be repeatedly using Lemma 1 in the derivation of $\mathbf{b}_\alpha^*$, as follows. Let $\mathbf{b}_{\alpha|i}(\mathbf{x}_\alpha|x_i)$ and $H(\mathbf{b}_{\alpha|i})$ be defined below:

$$\begin{aligned} \mathbf{b}_{\alpha|i}(\mathbf{x}_\alpha|x_i) &\overset{def}{=} \frac{b_\alpha(\mathbf{x}_\alpha)}{b_i(x_i)}\\ H(\mathbf{b}_{\alpha|i}) &\overset{def}{=} -\sum_{\mathbf{x}_\alpha\setminus x_i} \frac{b_\alpha(\mathbf{x}_\alpha)}{b_i(x_i)}\ln\frac{b_\alpha(\mathbf{x}_\alpha)}{b_i(x_i)}. \end{aligned}$$

Note that the constraint $\mathbf{b}_{\alpha|i}(\mathbf{x}_\alpha|x_i) \in \mathcal{P}$, i.e., that $\mathbf{b}_{\alpha|i}$ lives in the probability simplex, is equivalent to the marginal consistency constraint $\sum_{\mathbf{x}_\alpha\setminus x_i} b_\alpha(\mathbf{x}_\alpha) = \mathbf{b}(x_i)$ as well. We can use $H(\mathbf{b}_{\alpha|i})$ to simplify the conditional entropy term $H(\mathbf{b}_\alpha) - H(\mathbf{b}_i)$ by the following Lemma:

Lemma 2:

$$H(\mathbf{b}_\alpha) - H(\mathbf{b}_i) = \sum_{x_i} b_i(x_i)H(\mathbf{b}_{\alpha|i})$$

Proof: The Lemma is based on the definition of conditional entropy $H(X \mid Y) = H(X,Y) - H(Y) = \sum_y p(y)H(X \mid Y = y)$ for random variables X, Y. In our

$$\min_{b_i(x_i)\in\mathcal{P}}\left\{-\sum_{x_i} b_i(x_i)\ln\phi_i(x_i) - \epsilon\hat{c}_i H(\mathbf{b}_i) + \sum_{x_i} b_i(x_i) \underbrace{\sum_{\alpha\in N(i)}\epsilon\hat{c}_{i\alpha}\left[\underbrace{\min_{b_{\alpha|i}\in\mathcal{P}} - \sum_{\mathbf{x}_\alpha\setminus x_i}\mathbf{b}_{\alpha|i}(\mathbf{x}_\alpha|x_i)\ln\hat{\psi}_{i,\alpha}^{1/(\epsilon\hat{c}_{i\alpha})}(\mathbf{x}_\alpha) - H(\mathbf{b}_{\alpha|i})}_{(*)}\right]}_{(**)}\right\} \tag{26}$$

Fig. 8. Reducing the local sub-problem in Fig. 7 to a series of normalizations by introducing conditional entropies.

terms we have $H(\mathbf{x}_\alpha \setminus x_i \mid x_i) = H(\mathbf{b}_\alpha) - H(\mathbf{b}_i) = \sum_{x_i} b_i(x_i)H(\mathbf{b}_{\alpha|i})$. $\square$

With the definitions above, the optimization problem of step (b) as described in eqn. 25, can be broken down to a cascade of two steps, described in eqn. 26 in Fig. 8.

From Lemma 1 (eqn. 29) we obtain the solution for the inner optimization block $(*)$:

$$(*) = -\ln\sum_{\mathbf{x}_\alpha\setminus x_i}\hat{\psi}_{i,\alpha}(\mathbf{x}_\alpha)^{1/(\epsilon\hat{c}_{i\alpha})}.$$

We make the following definition:

$$m_{\alpha\to i}(x_i) \overset{def}{=} \left(\sum_{\mathbf{x}_\alpha\setminus x_i}\hat{\psi}_{i,\alpha}(\mathbf{x}_\alpha)^{1/(\epsilon\hat{c}_{i\alpha})}\right)^{\epsilon\hat{c}_{i\alpha}} \tag{30}$$

Therefore, the inner-block denoted by $(**)$ takes the form:

$$(**) = -\ln\prod_{\alpha\in N(i)} m_{\alpha\to i}(x_i).$$

Substituting $(**)$ back into eqn. 26 we obtain:

$$\min_{b_i(x_i)\in\mathcal{P}}\epsilon\hat{c}_i\left[-H(\mathbf{b}_i) - \sum_{x_i} b_i(x_i)\ln\phi_i^{1/\epsilon\hat{c}_i}(x_i)\prod_{\alpha\in N(i)} m_{\alpha\to i}^{1/\epsilon\hat{c}_i}(x_i)\right]$$

and from Lemma 1 (eqn. 28) we obtain a closed-form solution for $b_i^*(x_i)$:

$$b_i^*(x_i) \propto \left(\phi_i(x_i)\prod_{\alpha\in N(i)} m_{\alpha\to i}(x_i)\right)^{1/\epsilon\hat{c}_i}. \tag{31}$$

Finally, $b_\alpha^*(\mathbf{x}_\alpha) = \mathbf{b}_{\alpha|i}^*(\mathbf{x}_\alpha|x_i)b_i^*(x_i)$ takes the form:

$$b_\alpha^*(\mathbf{x}_\alpha) = \frac{b_i^*(x_i)}{m_{\alpha\to i}^{1/\epsilon\hat{c}_{i\alpha}}(x_i)}\hat{\psi}_{i,\alpha}(\mathbf{x}_\alpha)^{1/(\epsilon\hat{c}_{i\alpha})} \tag{32}$$

Next we evaluate step (c) of Algorithm 1, i.e.,

$$\boldsymbol{\lambda}_{i,\alpha}^*(\mathbf{x}_\alpha) = -\mu_{i,\alpha}(\mathbf{x}_\alpha) - \nabla\hat{f}_\epsilon(b_\alpha^*(\mathbf{x}_\alpha)) + \sigma_\alpha\mathbf{1}, \tag{33}$$

where σ_α is an arbitrary scalar and $\hat{f}_\epsilon$ is defined in eqn. 10. Define $n_{i\to\alpha}(\mathbf{x}_\alpha)$ as follows:

$$n_{i\to\alpha}(\mathbf{x}_\alpha) \overset{def}{=} \exp(-\lambda_{i,\alpha}(\mathbf{x}_\alpha)) \tag{34}$$

We note, therefore, that the additive constant freedom in the definition of $\boldsymbol{\lambda}_{i,\alpha}$ becomes a scaling choice in the definition of $n_{i\to\alpha}$. Without loss of generality we choose the scale such that $n_{i\to\alpha}(\mathbf{x}_\alpha)\in\mathcal{P}$. The claim below sets the value of $n_{i\to\alpha}$:

Proposition 2:

$$n_{i\to\alpha}(\mathbf{x}_\alpha) \propto \left(\frac{b_i^*(x_i)}{m_{\alpha\to i}^{1/\epsilon\hat{c}_{i\alpha}}(x_i)}\right)^{\epsilon c_\alpha}\hat{\psi}_{i,\alpha}(\mathbf{x}_\alpha)^{-c_{i\alpha}/\hat{c}_{i\alpha}} \tag{35}$$

Proof: From definition of $n_{i\to\alpha}$ and from eqn. 33 we have:

$$n_{i\to\alpha}(\mathbf{x}_\alpha) \propto \exp(\mu_{i,\alpha}(\mathbf{x}_\alpha))\exp(\nabla\hat{f}_\epsilon(b_\alpha^*(\mathbf{x}_\alpha))).$$

Substituting the value of $\nabla\hat{f}_\epsilon(b_\alpha^*(\mathbf{x}_\alpha))$:

$$\nabla\hat{f}_\epsilon(b_\alpha^*(\mathbf{x}_\alpha)) = -\ln\psi_\alpha(\mathbf{x}_\alpha) + \epsilon c_\alpha(\ln b_\alpha^*(\mathbf{x}_\alpha)+1),$$

and the value of $b_\alpha^*(\mathbf{x}_\alpha)$ from eqn. 32 we obtain:

$$\begin{aligned} n_{i\to\alpha}(\mathbf{x}_\alpha) &\propto \exp(\mu_{i,\alpha}(\mathbf{x}_\alpha) - \ln\psi_\alpha(\mathbf{x}_\alpha))(b_\alpha^*(\mathbf{x}_\alpha))^{\epsilon c_\alpha} \\ &= \hat{\psi}_{i,\alpha}^{-1}(\mathbf{x}_\alpha)\left(\frac{b_i^*(x_i)}{m_{\alpha\to i}^{1/\epsilon\hat{c}_{i\alpha}}(x_i)}\right)^{\epsilon c_\alpha}\hat{\psi}_{i,\alpha}^{c_\alpha/\hat{c}_{i\alpha}}(\mathbf{x}_\alpha) \\ &= \left(\frac{b_i^*(x_i)}{m_{\alpha\to i}^{1/\epsilon\hat{c}_{i\alpha}}(x_i)}\right)^{\epsilon c_\alpha}\hat{\psi}_{i,\alpha}^{c_\alpha/\hat{c}_{i\alpha}-1}(\mathbf{x}_\alpha) \end{aligned}$$

and following substitution of $\hat{c}_{i\alpha} = c_\alpha + c_{i\alpha}$ we obtain what we set out to prove. $\square$

Substituting $\boldsymbol{\mu}_{i,\alpha} = \sum_{j\in N(\alpha)\setminus i}\boldsymbol{\lambda}_{j,\alpha}$ into the definition of $\hat{\psi}_{i,\alpha}$ (eqn. 27) we obtain:

$$\begin{aligned} \hat{\psi}_{i,\alpha}(\mathbf{x}_\alpha) &= \psi_\alpha(\mathbf{x}_\alpha)\prod_{j\in N(\alpha)\setminus i}\exp(-\boldsymbol{\lambda}_{j,\alpha}(\mathbf{x}_\alpha)) \\ &= \psi_\alpha(\mathbf{x}_\alpha)\prod_{j\in N(\alpha)\setminus i} n_{j\to\alpha}(\mathbf{x}_\alpha) \end{aligned} \tag{36}$$

Substituting eqn. 36 into eqn. 30 we obtain the update rule for $m_{\alpha\to i}$:

$$m_{\alpha\to i}(x_i) = \left(\sum_{\mathbf{x}_\alpha\setminus x_i}\left(\psi_\alpha(\mathbf{x}_\alpha)\prod_{j\in N(\alpha)\setminus i} n_{j\to\alpha}(\mathbf{x}_\alpha)\right)^{1/\epsilon\hat{c}_{i\alpha}}\right)^{\epsilon\hat{c}_{i\alpha}}. \tag{37}$$

Substituting eqn. 36 into eqn. 35 we obtain:

$$n_{i\to\alpha}(\mathbf{x}_\alpha) \propto \left(\frac{b_i^*(x_i)}{m_{\alpha\to i}^{1/\epsilon\hat{c}_{i\alpha}}(x_i)}\right)^{\epsilon c_\alpha}\left(\psi_\alpha(\mathbf{x}_\alpha)\prod_{j\in N(\alpha)\setminus i} n_{j\to\alpha}(\mathbf{x}_\alpha)\right)^{\frac{-c_{i\alpha}}{\hat{c}_{i\alpha}}},$$

and substituting $\mathbf{b}_i^*$ in eqn. 31 we obtain the update rule for $n_{i\to\alpha}$.

APPENDIX D CONVEX-FREE-ENERGY PARAMETER SETTINGS

The fractional entropy approximation of eqn. 3

$$\sum_\alpha \bar{c}_\alpha H(\mathbf{b}_\alpha) + \sum_i \bar{c}_i H(\mathbf{b}_i),$$

is strictly convex if it can be written as eqn. 4

$$\sum_\alpha c_\alpha H(\mathbf{b}_\alpha) + \sum_i c_i H(\mathbf{b}_i) + \sum_{i,\alpha\in N(i)} c_{i\alpha}(H(\mathbf{b}_\alpha) - H(\mathbf{b}_i)),$$

in terms of $c_\alpha > 0, c_i, c_{i\alpha} \geq 0$. In this section we will introduce a number entropy approximations which fall into the convex-free-energy class. We will start with the Tree-re-weighted (TRW) entropy approximation [61] and then introduce other approximations.

There are two ways, introduced in the literature so far, to set parameters for the TRW entropy approximation — both of which do not belong the required setup of a convex-free-energy. In the first version, the TRW-free-energy corresponds to the setting of $c_\alpha > 0, c_i = 1 - \sum_{\alpha\in N(i)} c_\alpha$ and $c_{i\alpha} = 0$, where the setting of c_α corresponds to the relative number of spanning trees (or hyper-trees) of the graph which include the edge (hyperedge) α. The problem with this setting is that $c_i < 0$, thus, even though the fractional entropy approximation is convex, the functions h_i (defined in terms of c_i and $c_{i\alpha}$) are not convex.

The second version, introduced by [17], sets c_i as the relative number of spanning trees that have node i as a root, and for an edge $\alpha = (i,j)$, $c_{i\alpha}$ is the relative number of trees that include the directed edge $j \to i$. It is possible to find such edge probabilities for the uniform distribution over all spanning trees by employing a variant of the matrix tree theorem for directed trees, [17], [58] p.141. In this formulation $c_i, c_{i\alpha} > 0$ but $c_\alpha = 0$. The problem with $c_\alpha = 0$ is that the function f is no longer strictly convex.

In the claim below we show how to convert a TRW setting according to the second version, i.e., where $c_i, c_{i\alpha} > 0, c_\alpha = 0$ to the convex-free-energy setting $c'_\alpha > 0, c'_i, c'_{i\alpha} \geq 0$:

Claim 10: Assume an approximated entropy $\sum_\alpha \bar{c}_\alpha H(\mathbf{b}_\alpha) + \sum_i \bar{c}_i H(\mathbf{b}_i)$ is described by $c_i, c_{i\alpha} > 0$ and $c_\alpha = 0$, i.e. $\bar{c}_\alpha = c_\alpha + \sum_{i\in N(\alpha)} c_{i\alpha}$ and $\bar{c}_i = c_i - \sum_{\alpha\in N(i)} c_{i\alpha}$. Then there exists $c'_\alpha, c'_i, c'_{i\alpha} > 0$ which agree on the approximated entropy, namely $\bar{c}_\alpha = c'_\alpha + \sum_{i\in N(\alpha)} c'_{i\alpha}$ and $\bar{c}_i = c'_i - \sum_{\alpha\in N(i)} c'_{i\alpha}$

Proof: We describe an efficient algorithm for constructing the desired convex free energy: Initialize $c'_\alpha = 0, c'_i = 0, c'_{i\alpha} = 0$. For every $i = 1, ..., n$ and every $\alpha \in N(i)$ consider the entropy combination

$$c_{i\alpha}(H(\mathbf{b}_\alpha) - H(\mathbf{b}_i)) + \frac{c_i}{d_i} H(\mathbf{b}_i)$$

and divide it to two cases:

1) $c_{i\alpha} \leq c_i/d_i$, then the entropy can be equivalently written by the entropy

$$c_{i\alpha} H(\mathbf{b}_\alpha) + \left(\frac{c_i}{d_i} - c_{i\alpha}\right) H(\mathbf{b}_i)$$

therefore perform
   a) $c'_\alpha \leftarrow c'_\alpha + c_{i\alpha}$
   b) $c'_i \leftarrow c'_i + (\frac{c_i}{d_i} - c_{i\alpha})$.
2) $c_{i\alpha} > c_i/d_i$, then the entropy can be equivalently presented as:

$$\frac{c_i}{d_i} H(\mathbf{b}_\alpha) + \left(c_{i\alpha} - \frac{c_i}{d_i}\right)(H(\mathbf{b}_\alpha) - H(\mathbf{b}_i)),$$

therefore perform
   a) $c'_\alpha \leftarrow c'_\alpha + \frac{c_i}{d_i}$
   b) $c'_{i\alpha} \leftarrow c'_{i\alpha} + (c_{i\alpha} - \frac{c_i}{d_i})$.

Since $c_i, c_{i\alpha}$ are positive we obtain an equivalent entropy approximation with $c'_\alpha > 0$ and $c'_i, c'_{i\alpha} \geq 0$. A straight forward bookkeeping ensures that $\bar{c}_\alpha$ and $\bar{c}_i$ do not change.

$\square$

Another concave approximation is to seek a setting of parameters $c_\alpha > 0, c_i, c_{i\alpha} \geq 0$ such that the approximation $\tilde{H}$ is as close as possible to the non-convex Bethe approximation[2]. Given the equations in Definition 1 connecting the parameters $c_\alpha, c_i, c_{i\alpha}$ to $\bar{c}_\alpha$ and $\bar{c}_i$, the space of admissible solutions must satisfy the following equations:

$$c_i + \sum_{\alpha\in N(i)} (c_\alpha + \sum_{j\in N(\alpha)\setminus i} c_{j\alpha}) = 1, \quad i = 1, ..., n$$
$$c_i, c_{i\alpha} \geq 0, \quad c_\alpha > 0.$$

Among all possible admissible solutions we choose the one in which $\bar{c}_\alpha$ is as uniform as possible, i.e., we apply Laplace's principle of insufficient reasoning. The criterion function, therefore, minimizes:

$$\min_{c_i, c_{i\alpha}, c_\alpha \in admissible} \sum_\alpha (c_\alpha + \sum_{i\in N(\alpha)} c_{i\alpha} - 1)^2, \tag{38}$$

which is a least-squares criteria for uniformity of $\bar{c}_\alpha$. We refer to the two least-squares scheme as L_2 convex free energy approximation. In an earlier work [19], we also used the maximum entropy approach where the criterion function minimizes $\sum_\alpha \bar{c}_\alpha \ln \bar{c}_\alpha$. Further investigation for constructing good convex free energy approximations can be found in [39].

The desire towards uniformity, besides being used extensively in probabilistic settings, is motivated by the success of the Bethe free energy where $\bar{c}_\alpha = 1$. The Bethe free energy is non-convex for factor graphs with cycles, thus is not a member of the convex free energies, but empirical evidence suggest that when BP converges the marginals are surprisingly good. For Bethe free energy $\bar{c}_\alpha = 1$ over all factor nodes α — hence our proposal to strive for uniformity over the space of admissible solutions. In some sense we are attempting to "convexify" the Bethe free energy, although this is not being done directly.

[2]A similar idea was independently derived by Nir Friedman and his collaborators — personal communication.

APPENDIX E
INCORPORATING ZERO POTENTIALS

A particularly important class of factors are those with zero potentials. These type of potentials are used, for example, in defining error-correcting codes. Note that if one or more of the factor potentials $\psi_\alpha(x_\alpha)$ or local potentials $\phi(x_i)$ are equal to zero, then the overall probability of states which contain these configurations is zero, namely $p(\mathbf{x}_\alpha) = 0$ or $p(x_i) = 0$ respectively. This restriction on the marginal probabilities implicitly appears in the variational programs using the convention $0 \ln 0 = 0$ and $-x \ln 0 = \infty$ for $x > 0$.

Recall that the variational approach seeks a distribution $p(x_1, ..., x_n)$ which is as close as possible, in relative entropy terms, to the product $\prod_i \phi_i(x_i) \prod_\alpha \psi_\alpha(\mathbf{x}_\alpha)$. Expanding the relative entropy produces the free energy:

$$g(\mathbf{p}) = \sum_{\alpha, \mathbf{x}_\alpha} \theta_\alpha(\mathbf{x}_\alpha) p(\mathbf{x}_\alpha) + \sum_{i, x_i} \theta_i(x_i) p(x_i) - H(\mathbf{p}),$$

where $\theta_\alpha = -\ln \psi_\alpha$, $\theta_i = -\ln \phi_i$, and $p(\mathbf{x}_\alpha)$, $p(x_i)$ are the marginal probabilities, and $H(\mathbf{p})$ is the entropy function. Since $-x \ln 0 = \infty$ whenever $x > 0$, the zero potential $\psi_\alpha(\mathbf{x}_\alpha) = 0$ constraints $\mathbf{p} \in dom(g)$ if and only if $p(\mathbf{x}_\alpha) = 0$, Likewise, $\phi_i(x_i) = 0$ constrains $\mathbf{p} \in dom(g)$ whenever $p(x_i) = 0$. Following the above, the inference program which corresponds to the free energy minimization is well defined for zero potentials when we consider the domain of the free energy, and takes the form $\min_{\mathbf{p} \in dom(g)} g(\mathbf{p})$. In spite the mathematical difficulty introduced by using zero potentials, it makes no difference from algorithmic perspective, since the optimal distribution $\mathbf{p}^*$ is the normalization of the product of potentials, $\mathbf{p}^* \propto \prod_i \phi_i \prod_\alpha \psi_\alpha$ and it respects the domain constraint.

The same behavior appears in the variational approach for MAP assignment, where one seeks vector $\mathbf{x}^*$ which maximize the energy $\prod_i \phi_i(x_i) \prod_\alpha \psi_\alpha(\mathbf{x}_\alpha)$. This task is described by the linear function

$$g(\mathbf{p}) = \sum_{\alpha, x_\alpha} \theta_\alpha(\mathbf{x}_\alpha) p(\mathbf{x}_\alpha) + \sum_{i, x_i} \theta_i(x_i) p(x_i),$$

whereas the zero potentials of the form $\phi_i(x_i) = 0$ or $\psi_\alpha(\mathbf{x}_\alpha) = 0$ constraints $\mathbf{p} \in dom(g)$ if and only if $p(x_i) = 0$ or $p(\mathbf{x}_\alpha) = 0$ respectively. Again, this mathematical nuance makes no difference from algorithmic perspective, since the optimal distribution $\mathbf{p}^*$ is the a zero-one distribution, i.e. $p^*(\mathbf{x}^*) = 1$ and for every $\mathbf{x} \neq \mathbf{x}^*$ holds $p^*(\mathbf{x}) = 0$.

The framework of incorporating zero potentials in the domain of the optimized program also corresponds to the approximate inference and LP-relazation described by the minimization of the function

$$g(\mathbf{b}) = \sum_{i, x_i} \theta_i(x_i) b_i(x_i) + \sum_{\alpha, \mathbf{x}_\alpha} \theta_\alpha(\mathbf{x}_\alpha) b_\alpha(\mathbf{x}_\alpha) - \epsilon \tilde{H}(\mathbf{b}),$$

whose domain is constrained by zero potentials, namely $\phi_i(x_i) = 0$ or $\psi_\alpha(\mathbf{x}_\alpha) = 0$ constraints $\mathbf{b} \in dom(g)$ if and only if $b_i(x_i) = 0$ or $b_\alpha(\mathbf{x}_\alpha) = 0$ respectively. The domain constrains are inherited by the norm-product algorithm where we represent $g(\mathbf{b})$ in the form $f_\epsilon(\mathbf{b}) + \sum_i h_{\epsilon,i}(\mathbf{b})$ described in eqns. 10, 11. In particular, $\mathbf{b} \in dom(f_\epsilon)$ only if $b_\alpha(\mathbf{x}_\alpha) = 0$ whenever $\psi_\alpha(\mathbf{x}_\alpha) = 0$, and $\mathbf{b} \in dom(h_{\epsilon,i})$ only if $b_i(x_i) = 0$ whenever $\phi_i(x_i) = 0$. This domain constraint do not affect the norm-product algorithm whose optimal beliefs are a (power) normalization of the potentials multiplied by the messages, described in eqn. 31 and eqn. 32. Therefore, in the norm-product optimization framework, zero potential $\psi_\alpha(\mathbf{x}_\alpha) = 0$ or $\phi_i(x_i) = 0$ induces optimal beliefs satisfying $b^*_\alpha(\mathbf{x}_\alpha) = 0$ or $b^*_i(x_i) = 0$ respectively.

APPENDIX F
ACKNOWLEDGEMENTS

The authors wish to Chen Yanover, Talya Meltzer, Ofer Meshi, Ariel Jaimovich, Amir Globerson and Yair Weiss for helpful discussions. We also thank the anonymous referees for their criticisms and suggestions.

REFERENCES

[1] R. Baxter, *Exactly Solved Models in Statistical Mechanics*. Academic Press, 1982.

[2] M. Bayati, D. Shah, and M. Sharma, "Maximum weight matching via max-product belief propagation," in *Information Theory, 2005. ISIT 2005. Proceedings. International Symposium on*, 2005, pp. 1763–1767.

[3] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near shannon limit error-correcting coding and decoding: Turbo-codes," in *IEEE International Conference on Communications*, Geneva, Switzerland, 1993, pp. 1064–1070.

[4] D. P. Bertsekas, A. Nedić, and A. E. Ozdaglar, *Convex Analysis and Optimization*. Athena Scientific, 2003.

[5] L. Bregman, "The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming," *U.S.S.R Comput. Math. and Math. Physics*, vol. 7, pp. 200–217, 1967.

[6] L. Bregman, Y. Censor, and S. Reich, "Dykstras algorithm as the nonlinear extension of Bregmans optimization method," *Journal of Convex Analysis*, vol. 6, no. 2, pp. 319–333, 1999.

[7] Y. Censor and S. Reich, "The Dykstra algorithm with Bregman projections," *Communications in Applied Analysis*, vol. 2, no. 3, pp. 407–419, 1998.

[8] M. Chertkov and V. Chernyak, "Loop series for discrete statistical models on graphs," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2006, p. P06009, 2006.

[9] T. Cho, M. Butman, S. Avidan, and W. Freeman, "The patch transform and its applications to image editing," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, 2008, pp. 1–8.

[10] I. Csiszar, "I-divergence geometry of probability distributions and minimization problems," *The Annals of Probability*, vol. 3, no. 1, pp. 146–158, 1975.

[11] R. Dykstra, "An iterative procedure for obtaining i-projections onto the intersection of convex sets," *The Annals of Probability*, vol. 13, pp. 975–984, 1985.

[12] ——, "An algorithm for restricted least squares regression," *J. of the Amer. Stat. Assoc.*, vol. 78, pp. 837–842, 1983.

[13] J. Feldman, D. Karger, and M. Wainwright, "LP Decoding," in *proceedings of the annual Allerton conference on communication control and computing*, vol. 41. The University; 1998, 2003, pp. 951–960.

[14] W. Freeman, E. Pasztor, and O. Carmichael, "Learning Low-Level Vision," *International Journal of Computer Vision*, vol. 40, no. 1, pp. 25–47, 2000.

[15] B. Frey, *Graphical Models for Machine Learning and Digital Communications*. MIT Press, 1998.

[16] R. Gallager, *Low-density parity check codes*. MIT Press, 1963.

[17] A. Globerson and T. Jaakkola, "Convergent propagation algorithms via oriented trees," in *Conference on Uncertainty in Artifical Intelligence (UAI)*, 2007.

[18] ——, "Fixing max-product: convergent message passing algorithms for MAP relaxations," in *Proceedings of the conference on Neural Information Processing Systems (NIPS)*, 2007.

[19] T. Hazan and A. Shashua, "Convergent message-passing algorithms for inference over general graphs with convex free energies," in *Conference on Uncertainty in Artifical Intelligence (UAI)*, Helsinki, Finland, July 2008.

[20] T. Heskes, "On the Uniqueness of Loopy Belief Propagation Fixed Points," *Neural Computation*, vol. 16, no. 11, pp. 2379–2413, 2004.

[21] ——, "On the Uniqueness of Loopy Belief Propagation Fixed Points," *Neural Computation*, vol. 16, no. 11, pp. 2379–2413, 2004.

[22] ——, "Convexity Arguments for Efficient Minimization of the Bethe and Kikuchi Free Energies," *Journal of Artificial Intelligence Research*, vol. 26, pp. 153–190, 2006.

[23] C. Hildreth, "A quadratic programming procedure," *Naval Research Logistics Quarterly*, 1957.

[24] B. Huang and T. Jebara, "Loopy belief propagation for bipartite maximum weight b-matching," *Artificial Intelligence and Statistics (AISTATS)*, 2007.

[25] T. Jaakkola and M. Jordan, "Variational probabilistic inference and the QMR-DT database," *Journal of Artificial Intelligence Research*, 1999.

[26] J. Johnson, D. Malioutov, and A. Willsky, "Lagrangian relaxation for MAP estimation in graphical models," in *Proceedings of the Allerton Conference on Control, Communication and Computing*. Citeseer, 2007.

[27] M. Jordan, *Learning in Graphical Models*. MIT Press, 1998.

[28] R. Kalman, "A new approach to linear filtering and prediction problems," *J. Basic Eng.*, vol. 82, pp. 34–45, 1960.

[29] R. Kikuchi, "A theory of cooperative phenomena," *Physical Review*, vol. 81, no. 6, pp. 988–1003, 1951.

[30] V. Kolmogorov, "Convergent tree-reweighted message passing for energy minimization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 10, pp. 1568–1583, 2006.

[31] V. Kolmogorov and M. Wainwright, "On the optimality of tree-reweighted max-product message passing," in *Uncertainty in Artificial Intelligence*. Citeseer, 2005.

[32] N. Komodakis and N. Paragios, "Beyond loose LP-relaxations: Optimizing MRFs by repairing cycles," *Computer Vision–ECCV 2008*, pp. 806–820, 2008.

[33] N. Komodakis, N. Paragios, and G. Tziritas, "MRF Energy Minimization & Beyond via Dual Decomposition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, In Press.

[34] A. Koster, S. Hoesel, and A. Kolen, "The partial constraint satisfaction problem: Facets and lifting theorems," *Operations Research Letters*, vol. 23, no. 3-5, pp. 89–97, 1998.

[35] F. Kschischang, B. Frey, and H. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 498–519, 2001.

[36] Z. Luo and P. Tseng, "On the convergence rate of dual ascent methods for linearly constrained convex minimization," *Mathematics of Operations Research*, vol. 18, no. 4, pp. 846–867, 1993.

[37] T. Meltzer, A. Globerson, and Y. Weiss, "Convergent message passing algorithms-a unifying view," in *In Uncertainty in Artificial Intelligence (UAI)*, 2009.

[38] T. Meltzer, C. Yanover, and Y. Weiss, "Globally optimal solutions for energy minimization in stereo vision using reweighted belief propagation," in *Proc. Int. Conf. Comp. Vision*, 2005.

[39] O. Meshi, A. Jaimovich, A. Globerson, and N. Friedman, "Convexifying the bethe free energy," in *Conference on Uncertainty in Artifical Intelligence (UAI)*, 2009.

[40] J. Mooij, "libDAI: A free/open source C++ library for discrete approximate inference methods," 2009, http://www.libdai.org.

[41] J. Mooij and H. Kappen, "Sufficient conditions for convergence of loopy belief propagation," in *Conference on Uncertainty in Artifical Intelligence (UAI)*, 2005.

[42] Y. Nesterov, *Introductory lectures on convex optimization: A basic course*. Springer Netherlands, 2004.

[43] J. V. Neumann, *Functional Operators Vol. II*. Princeton University Press, 1950.

[44] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, 1988.

[45] P. Ravikumar, A. Agarwal, and M. Wainwright, "Message-passing for graph-structured linear programs: proximal projections, convergence and rounding schemes," in *Proceedings of the 25th international conference on Machine learning*. ACM New York, NY, USA, 2008, pp. 800–807.

[46] R. Rockafellar, *Convex Analysis*. Princeton University Press, 1970.

[47] S. Sanghavi, D. Malioutov, and A. Willsky, "Linear programming analysis of loopy belief propagation for weighted matching," *Advances in Neural Information Processing Systems*, 2007.

[48] M. Schlesinger, "Syntactic analysis of two-dimensional visual signals in noisy conditions," *Kibernetika, Kiev*, vol. 4, pp. 113–130, 1976.

[49] N. Shental, A. Zomet, T. Hertz, and Y. Weiss, "Learning and Inferring Image Segmentations using the GBP Typical Cut Algorithm," in *International Conference on Computer Vision (ICCV)*, 2003.

[50] S. Shimony, "Finding MAPs for belief networks is NP-hard," *Artificial Intelligence*, vol. 68, no. 2, pp. 399–410, 1994.

[51] D. Sontag and T. Jaakkola, "New outer bounds on the marginal polytope," in *Advances in Neural Information Processing Systems (NIPS)*. Cambridge, MA: MIT Press, 2008, pp. 1393–1400.

[52] ——, "Tree block coordinate descent for MAP in graphical models," in *12th International Workshop on Artificial Intelligence and Statistics (AI-STATS)*, 2009.

[53] D. Sontag, T. Meltzer, A. Globerson, T. Jaakkola, and Y. Weiss, "Tightening lp relaxations for map using message passing," in *In Uncertainty in Artificial Intelligence (UAI)*, 2008.

[54] E. Sudderth, M. Wainwright, and A. Willsky, "Loop series and Bethe variational bounds in attractive graphical models," *Advances in neural information processing systems*, vol. 20, pp. 1425–1432, 2008.

[55] M. Tappen and W. Freeman, "Comparison of graph cuts with belief propagation for stereo, using identical MRF parameters," in *IEEE International Conference on Computer Vision*, vol. 2, 2003, pp. 900–906.

[56] S. Tatikonda and M. Jordan, "Loopy belief propagation and Gibbs measures," in *In Uncertainty in Artificial Intelligence*, 2002.

[57] P. Tseng and S. Yun, "A coordinate gradient descent method for nonsmooth separable minimization," *Mathematical Programming*, vol. 117, no. 1, pp. 387–423, 2009.

[58] W. Tutte, *Graph theory*. Cambridge University Press, 2001.

[59] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE trans. on Information Theory*, vol. 13, no. 2, pp. 260–269, 1967.

[60] P. Vontobel and R. Koetter, "Towards low-complexity linear-programming decoding," *Arxiv preprint cs/0602088*, 2006.

[61] M. J. Wainwright, T. Jaakkola, and A. S. Willsky, "Tree-based reparameterization for approximate estimation on graphs with cycles," in *Proceedings of the conference on Neural Information Processing Systems (NIPS)*, Vancouver, Canada, December 2002.

[62] M. Wainwright, T. Jaakkola, and A. Willsky, "A new class of upper bounds on the log partition function," *Information Theory, IEEE Transactions on*, vol. 51, no. 7, pp. 2313–2335, 2005.

[63] ——, "MAP estimation via agreement on trees: message-passing and linear programming," *IEEE Transactions on Information Theory*, vol. 51, no. 11, pp. 3697–3717, 2005.

[64] Y. Weiss, "Correctness of local probability propagation in graphical models with loops," *Neural Computation*, vol. 12, no. 1, pp. 1–41, 2000.

[65] Y. Weiss, C. Yanover, and T. Meltzer, "Map estimation, linear programming and belief propagation with convex free energies," in *Uncertainty in Artifical Inteligence, Proceedings of the nineteenth conference (UAI)*, 2007.

[66] T. Werner, "A linear programming approach to max-sum problem: A review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 7, pp. 1165–1179, 2007.

[67] C. Yanover, T. Meltzer, and Y. Weiss, "Linear Programming Relaxations and Belief Propagation–An Empirical Study," *The Journal of Machine Learning Research*, vol. 7, p. 1907, 2006.

[68] C. Yanover and Y. Weiss, "Approximate Inference and Protein-Folding," *Proceedings of the conference on Neural Information Processing Systems (NIPS)*, pp. 1481–1488, 2003.

[69] J. Yedidia, W. Freeman, and Y. Weiss, "Constructing free-energy approximations and generalized belief propagation algorithms," *Information Theory, IEEE Transactions on*, vol. 51, no. 7, pp. 2282–2312, 2005.

[70] A. Yuille, "CCCP Algorithms to Minimize the Bethe and Kikuchi Free Energies: Convergent Alternatives to Belief Propagation," *Neural Computation*, vol. 14, no. 7, pp. 1691–1722, 2002.